BÁO CÁO CÁC VẤN ĐỀ HIỆN ĐẠI TRONG CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI: TÌM HIỂU VỀ ETHEREUM

Giảng viên: PGS.TS Trương Anh Hoàng

Nhóm 8

Thành viên:

Nguyễn Mạnh Cường

Phạm Minh Đức

Bùi Thị Trung Thủy

Vũ Nam Tước

Mục lục

1 Đặt vấn đề

Trên thế giới hiện nay, nổi lên các hoạt động mua bán, trao đổi tiền ảo rất sôi nổi, trong đó Bitcoin và Ethereum (chúng ta sẽ tìm hiểu bên dưới) là 2 đồng tiền được nhắc đến nhiều nhất, xoay quanh những chủ đề này nhiều người còn quan tâm tới chuỗi khối - một công nghệ đằng sau những tiền ảo này. Công nghệ này đang giúp mọi người dù xa lạ hay thân quen đều có thể xây dựng một cuốn sổ giao dịch đáng tin cậy cho riêng mình. Đây là yếu tố có ý nghĩa vượt xa cả mọi giao thức mã hóa tiền tệ thông thường.

Với Ethereum, dù sinh sau đẻ muộn nhưng nó cho thấy được những ưu điểm vượt trội hơn Bitcoin về nhiều mặt, người ta hay ví von rằng Ethereum là phiên bản Bitcoin 2.0. Chúng ta sẽ cùng tìm hiểu kỹ hơn về chuối khối, về Ethereum và tiềm năng của chúng.

2 Tổng quan về chuỗi khối

2.1 Khái niệm chuỗi khối

Chuỗi khối (blockchain)[1] là một cơ sở dữ liệu phân cấp lưu trữ thông tin trong các khối thông tin được liên kết với nhau bằng mã hóa và mở rộng theo thời gian. Mỗi khối thông tin đều chứa thông tin về thời gian khởi tạo và được liên kết tới khối trước đó, kèm một mã thời gian và dữ liệu giao dịch. Chuỗi khối được thiết kế để chống lại việc thay đổi của dữ liệu: Một khi dữ liệu đã được mạng lưới chấp nhận thì sẽ không có cách nào thay đổi được nó. [2]

2.2 Các thành phần công nghệ của chuỗi khối

Ba thành phần công nghệ của chuỗi khối:

- Mạng ngang hàng: Một nhóm các máy tính có khả năng giao tiếp với nhau mà không phải phụ thuộc vào một người cầm quyền ở trung tâm và vì vậy không xảy ra hiện tượng điểm lỗi chí tử (single point of failure).
- Mật mã bất đối xứng: Một cách cho phép những máy tính này gửi các tin nhắn được mã hóa cho những người nhận đã được xác định vì vậy bất kỳ ai cũng có thể biết định danh của người gửi, nhưng chỉ người nhận được chỉ định mới có thể đọc nội dung tin nhắn. Ở Bitcoin[3] và Ethereum, mật mã bất đối xứng được sử dụng để tạo một tập các giấy chứng nhận (credential) cho tài khoản của bạn, để chắc chắn rảng chỉ có duy nhất bạn mới có thể chuyển các token của bạn (tiền của bạn).
- Phép băm mật mã: Một cách để sinh một "dấu-vân-tay" (fingerprint) nhỏ, duy nhất cho bất kỳ dữ liệu nào, cho phép so sánh một cách nhanh chóng các tập dữ liệu lớn và là một cách an toàn để xác nhận rằng dữ liệu đã được thay đổi hay chưa; ở cả Bitcoin và Ethereum, cấu trúc dữ liệu cây Merkle được sử dụng để ghi lại thứ tự kinh điển (canonical) của các giao dịch, sau đó được băm vào một “dấu-vân-tay” làm cơ sở cho việc so sánh của các máy tính trong mạng.

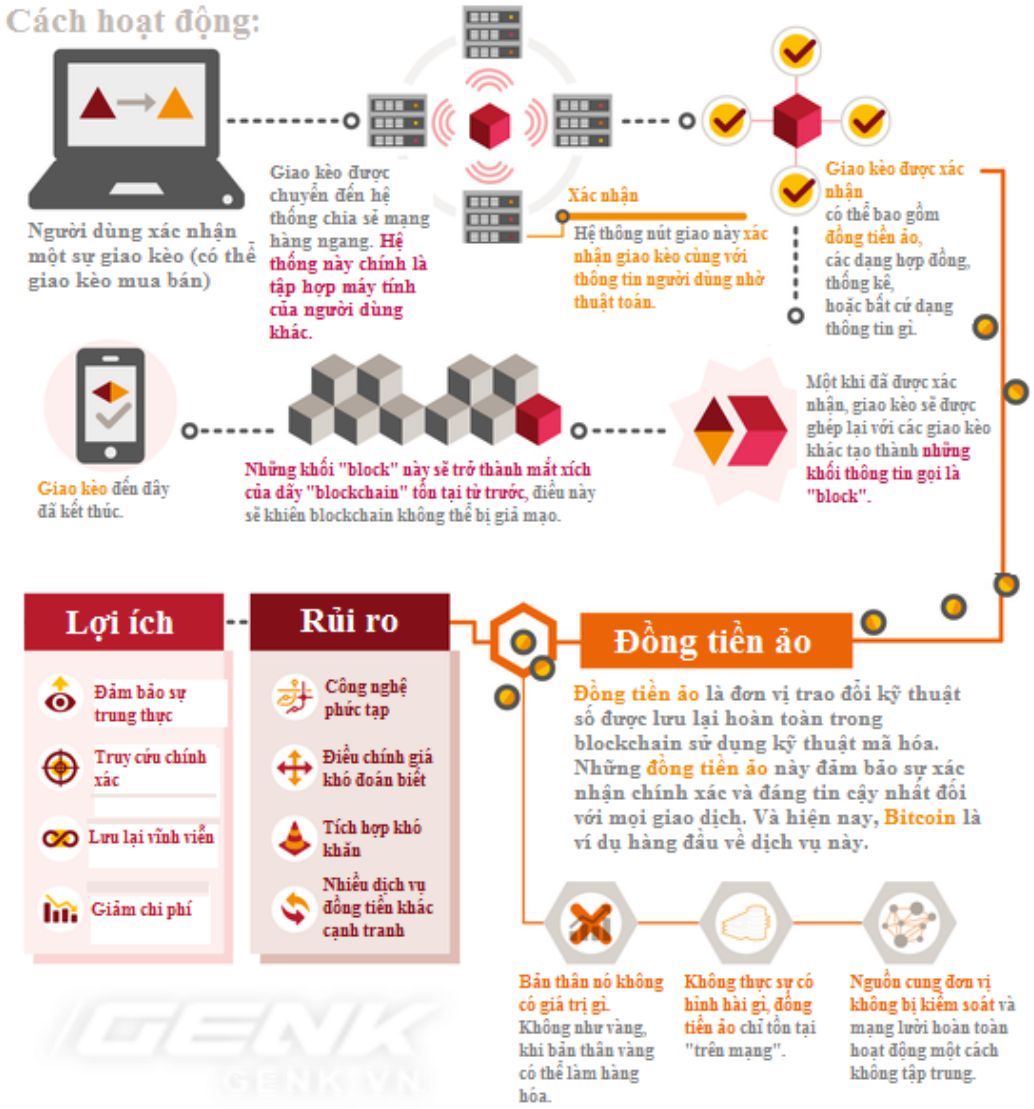
Cấu trúc của một khối (block):

- Index: thứ tự của block trong chuỗi (chain)
- Hash: Giá trị đại diện cho khối đó. Có đầu vào là Index, Previos Hash, Data, Timestamp và Nonce
- Previos Hash: giá trị của khối hợp lệ ngay trước
- Timestamp: thời gian khối được khai thác
- Data: dữ liệu được lưu trong khối đó
- Nonce: là giá trị được tìm thấy và thêm vào để sau khi băm, ta có giá trị băm hợp lệ.

2.3 Cách hoạt động của chuỗi khối

Ở chuỗi khối có những tính năng rất đặc biệt đó là có thể truyền tải dữ liệu mà không thông qua trung gian để xác nhận thông tin. Hệ thống chuỗi khối tồn tại nhiều nút độc lập với khả năng xác nhận thông tin. Mọi thông tin trong chuỗi khối có thể thay đổi hoặc bổ sung thêm khi có sự chấp nhận của tất cả các nút trên hệ thống. Hệ thống này có độ bảo mật cực cao mọi hành động đánh cắp dữ liệu đều sẽ bị chặn đứng.

Chuỗi khối thậm chí vẫn hoạt động bình thường khi một phần của hệ thống sụp đổ, những máy tính và các nút vẫn hoạt động để bảo vệ thông tin, giữ cho chuỗi khối không bị mất dữ liệu.



Hình 1: Minh hoạ cách hoạt động của chuỗi khối

3 Tổng quan về Ethereum

3.1 Giới thiệu chung

Khái niệm "Ethereum"[4] có thể được sử dụng để chỉ ba thứ khác nhau: giao thức Ethereum, mạng Ethereum, và dự án Ethereum bao gồm cả hai thứ trên. Có thể thấy sức mạnh chủ yếu của nó đến từ giao thức.

Ethereum là một nền tảng điện toán có tính chất phân tán, công cộng, mã nguồn mở dựa trên công nghệ Blockchain. Nó có tính năng Hợp đồng thông minh (Smart Contracts), tạo thuận lợi cho các thỏa thuận hợp đồng trực tuyến.

3.1.1 Lịch sử ra đời

Hình 2: Logo Ethereum

Ethereum ban đầu được mô tả trong một văn bản của Vitalik Buterin, một lập trình viên liên quan đến Bitcoin vào cuối năm 2013 với mục tiêu xây dựng các ứng dụng phân quyền. Buterin đã lập luận rằng Bitcoin cần một ngôn ngữ kịch bản để phát triển ứng dụng. Không đạt được thỏa thuận với nhóm phát triển Bitcoin, ông đề xuất phát triển một nền tảng mới với một ngôn ngữ kịch bản tổng quát hơn.

Bốn thành viên ban đầu của nhóm Ethereum là Vitalik Buterin, Mihai Alisie, Anthony Di Iorio và Charles Hoskinson. Phát triển chính thức của dự án phần mềm Ethereum bắt đầu vào đầu năm 2014 thông qua một công ty Thụy Sĩ tên là Ethereum Switzerland GmbH (EthSuisse). Sau đó, một tổ chức phi lợi nhuận tại Thụy Sĩ với tên gọi là Ethereum Foundation cũng được thành lập. Việc phát triển Ethereum được tài trợ bởi đám đông trực tuyến trong suốt tháng 7 và tháng 8 năm 2014, với những người tham gia mua Ethereum bằng các loại tiền kỹ thuật số khác như bitcoin. Mặc dù đã có những lời khen ngợi đầu tiên về những đổi mới kỹ thuật của Ethereum, nhưng cũng có các ngờ vực về tính an toàn và khả năng mở rộng của nó

3.2 Các thành phần cơ bản

3.2.1 Ether

Tiền mã hóa được giao dịch trong mạng lưới Ethereum được gọi là ether. Nó được liệt kê dưới mã ETH và giao dịch trên các sàn giao dịch tiền mã hóa. Nó cũng được sử dụng để trả phí giao dịch và dịch vụ tính toán trên mạng Ethereum.

Đối với tiền ảo, thách thức để được chấp nhận vẫn còn tồn tại. Ngày nay, những token này vẫn là một lớp thanh toán nhanh, bảo mật và minh bạch nhất trong các hệ thống tiền tệ được công nhận đang tồn tại; Một sự triển khai thử nghiệm một ngày nào đó có thể thay thế các công nghệ mạng thanh toán tập trung như Visa hay Mastercard như ngày nay.

3.2.2 Gas

Gas là một đơn vị công việc được sử dụng để đo lường mức chi phí tính toán (computationally expensive) cho một giao dịch trên Ethereum có thể tiêu tốn. Giá trị của Gas được trả bằng một lượng nhỏ ether.

Có hai lý do chính để Gas được ra đời:

- Thứ nhất, nó đảm bảo một phần thưởng được tả trước cho các thợ đào (miner) cho việc thực thi mã nguồn và bảo mật kết mạng, ngay cả khi việc thực thi bị thất bại vì một lý do nào đó.
- Thứ hai, nó hoạt động xung quanh bài toán rừng và đảm bảo việc thực thi không thể dài quá thời gian đã được ước lượng trước đó.

Gas là một đơn vị công việc, nó không phải là một đơn vị tiền tệ con, và bạn không thể sở hữu hay tích trữ nó. Nó chỉ đơn giản đo đạc mức tiêu hao mà hệ thống phải chịu nếu thực hiện giao dịch, ở mức các khái niệm tính toán. Để có thể trả chi phí Gas, bạn chỉ cần thêm ether vào tài khoản của bạn. Bạn không cần phải thu lấy nó một cách riêng biệt; không có khái niệm Gas token. Mọi toán tử trên EVM đều có một giá trị Gas nhất định.

Việc này khác so với ở Bitcoin, nơi mà chi phí được tính bằng kích thước của giao dịch tính bằng kilobytes, việc tính phí dựa trên khối lượng tính toán hợp lý hơn nhiều.

3.2.3 Máy ảo Ethereum (EVM)

Máy ảo Ethereum (EVM) là một môi trường chạy các hợp đồng thông minh Ethereum. Định nghĩa chính thức của EVM được quy định trong Ethereum Yellow Paper của Gavin Wood. Nó được hoàn toàn cô lập từ mạng, hệ thống tập tin và các quá trình khác của hệ thống máy chủ. Mỗi nút Ethereum trong mạng chạy một EVM và thực hiện các hướng dẫn giống nhau. Ethereum Virtual Machines đã được lập trình trong C++, Go, Haskell, Java, Python, Ruby, Rust và WebAssembly (hiện đang được phát triển)

3.2.4 Hợp đồng thông minh

Hợp đồng thông minh (smart contract) là một cơ chế trao đổi xác định, được kiểm soát bởi các phương tiện kỹ thuật số mà có thể giúp cho việc thực hiện giao dịch trực tiếp giữa các thực thể mà không cần tin cậy nhau. Các hợp đồng này được định nghĩa bằng cách lập trình và được chạy chính xác như mong muốn mà không bị kiểm duyệt, lừa đảo hay sự can thiệp từ bên thứ ba trung gian.

Chúng có thể được sử dụng để tạo điều kiện, xác minh và thực thi việc đàm phán hoặc thực hiện các hướng dẫn thủ tục kinh tế và có khả năng tránh được sự kiểm duyệt, thông đồng và rủi ro từ phía đối tác. Trong Ethereum, các hợp đồng thông minh được coi là các kịch bản tự trị hoặc các ứng dụng phân cấp được lưu trữ trong chuỗi khối Ethereum để thực hiện sau đó bởi EVM. Các hướng dẫn được nhúng trong các hợp đồng Ethereum được thanh toán bằng ether và có thể được thực hiện bằng nhiều ngôn ngữ Turing-complete khác nhau.

Sự khác biệt giữa hợp đồng truyền thống và hợp đồng hiện đại:

- Hợp đồng truyền thống được tạo bởi các chuyên gia pháp lí để biên soạn và thực thi. Điều này rất mất thời gian và không minh bạch. Hợp đồng có sự cố sảy ra thì phải dựa vào hệ thống pháp lí giải quyết và điều này tốn các chi phí liên quan.
- Hợp đồng thông minh được tạo ra bởi hệ thống máy tính bằng ngôn ngữ lập trình. Trong đó nêu rõ các điều khoản và hình phạt tương đương với các hợp đồng truyền thống đưa ra. Nhưng hợp đồng thông minh không cần sự can thiệp của con người, do đó đảm bảo việc thự thi hợp đồng được chính xác và công minh nhất. Toàn bộ đoàn mã của hợp đồng thông minh được thực hiện bởi hệ thống sổ cái phân tán chuỗi khối.

3.2.5 Solidity

Solidity là một ngôn ngữ lập trình sử dụng để viết các chương trình được gọi là hợp đồng thông minh (smart contract), thứ mà sẽ được chạy trên EVM. Ngôn ngữ mới này là một sự pha trộn các công ước từ mạng, hợp ngữ (assembly language) và phát triển web. Solidity là một ngôn ngữ cấp cao có định hướng hợp đồng, có cú pháp tương tự như của JavaScript và nó được thiết kế để nhắm mục tiêu Ethereum Virtual Machine (EVM).

Solidity là ngôn ngữ bậc cao được biên dịch bằng bytecode và được đưa lên chuỗi khối Ethereum bằng cách dùng các phần mềm phía người dùng như trình duyệt Mist hoặc nút (node). Ngôn ngữ Solidity định kiểu mạnh, hỗ trợ thừa kế, thư viện và các loại người dùng xác định và phức tạp giữa các tính năng khác.

Solidity có khả năng thể hiện tất cả các nhiệm vụ có thể thực hiện được bằng máy tính, làm cho chúng về mặt lý thuyết là Turing hoàn chỉnh. Điều đó có nghĩa là toàn bộ mạng phân phối, mỗi nút, thực hiện mọi chương trình được thực hiện trên nền tảng này. Khi một người dùng tải lên một hợp đồng thông minh thông qua nút Ethereum của họ, nó được bao gồm trong khối mới nhất và được truyền bá xung quanh mạng, nơi nó được lưu trữ trên mọi nút khác trong mạng.

3.2.6 Decenterlized App

Decenterlize App (DAPP) là ứng dụng mã nguồn mở hoàn chỉnh, hoạt động độc lập, không có thực thể nào kiểm soát phần lớn các tokens của ứng dụng này và dữ liệu và hồ sơ hoạt động của ứng dụng phải được lưu trữ dưới dạng mã hoá công khai, phân quyền chuỗi khối.

Ứng dụng này tạo các mã thông báo theo một thuật toán tiêu chuẩn hoặc bộ tiêu chuẩn và có thể phân phối một số hoặc tất cả các tokens của nó khi bắt đầu hoạt động. Các tokens này phải là cần thiết cho việc sử dụng ứng dụng và bất kỳ đóng góp nào từ người dùng sẽ được thưởng bằng các khoản thanh toán trong các tokens của ứng dụng. Ứng dụng có thể thích ứng các giao thức của nó để đáp ứng với những cải tiến đề xuất và phản hồi thị trường nhưng tất cả thay đổi phải được quyết định bởi sự đồng thuận của người sử dụng. DAPP có giá trị và có khả năng phá vỡ một số ngành công nghiệp.

Phân loại DAPP:

- Các ứng dụng phân cấp có chuỗi khối riêng như bitcoin.
- Giống như loại 1 nhưng có thêm token và các giao thức cần thiết như giao thức omni
- Giống như loại 2 nhưng có các mã thông báo cần thiết

Lợi ích của DAPP:

- The Safe Network: có sự bảo mật lớn nhất
- Factom: đơn giản hóa quá trình tăng cường quá trình ghi lại quá trình quản lý dữ liệu lớn
- BURST: giải quyết các vấn đề đã làm lưu trữ đám mây đắt đỏ và không đáng tin cậy
- Augur: được biết đến như là một thị trường dự đoán phân quyền để thưởng cho người sử dụng về các sự kiện dự báo
- BlockAuth: cung cấp một hệ thống đa chữ ký để chia sẻ dữ liệu cá nhân với các bên thứ ba

3.2.7 Giao thức GHOST (Greedy Heaviest Observed Subtree)

"Grey Heaviest Observed Subtree" (GHOST) là một cải tiến được giới thiệu lần đầu tiên vào tháng 12 năm 2013. Động lực đằng sau GHOST ra đời đó là thời gian xác nhận các chuỗi khối nhanh chóng hiện đang phải chịu sự giảm an ninh do tỷ lệ lỗi cao - bởi vì các khối mất một thời gian nhất định để truyền thông qua mạng.

Nếu thợ đào mỏ A khai thác một khối và sau đó thợ đào mỏ B tình cờ đào mỏ khác trước khi khối của thợ mỏ A truyền cho B, khối thợ mỏ B đào sẽ kết thúc, bị lãng phí và không đóng góp cho an ninh mạng. Hơn nữa, có một vấn đề tập trung hóa: nếu thợ mỏ A là một mỏ khai thác với 30% năng lượng (hashpower) và B có 10% năng lượng, A sẽ có nguy cơ tạo ra một khối cũ 70% thời gian (do đó 30% thời gian khác A sản xuất khối cuối cùng và vì vậy sẽ lấy được dữ liệu khai

thác được ngay lập tức) trong khi B sẽ có nguy cơ sản xuất một khối thời gian 90% thời gian. Do đó, nếu khoảng thời gian là đủ ngắn cho tỷ lệ khối cũ để được cao, A sẽ được hiệu quả hơn rất nhiều chỉ đơn giản bởi hiệu quả của kích thước của nó.

Với hai hiệu ứng kết hợp, các chuỗi khối sản xuất khối một cách nhanh chóng rất có thể dẫn đến một mở chứa có một tỷ lệ phần trăm đủ lớn của mạng lưới hashpower để kiểm soát quá trình khai thác trong thực tế. Như được mô tả, GHOST giải quyết vấn đề đầu tiên của mất mát an ninh mạng bằng cách bao gồm các khối cũ trong việc tính chuỗi nào là "dài nhất"; có nghĩa là không chỉ là cha mẹ và các tổ tiên khác của một khối, mà còn là hậu duệ cũ của tổ tiên của khối (trong thuật ngữ Ethereum, "Uncle" (bác)) được thêm vào để tính khối mà có tổng số bằng chứng của sự hỗ trợ công việc lớn nhất. Để giải quyết vấn đề thứ hai về xu hướng tập trung, chúng ta vượt qua các giao thức được mô tả, và cũng cung cấp các phần thưởng block cho các stales: một khối cũ nhận được 87,5% phần thưởng cơ bản của nó, và con cháu (nephew) có khối cũ nhận được 12,5% phần thưởng còn lại. Tuy nhiên, phí giao dịch không được thưởng cho Uncle.

Phiên bản Ethereum của Ghost chỉ rơi vào bảy cấp độ - hoặc trở lại bảy cấp độ trong chiều cao của chuỗi khối:

- Một khối phải chỉ định cha mẹ, và nó phải chỉ định 0 hoặc nhiều Uncle
- Đối với mỗi Uncle U trong khối B, thợ mỏ B được cộng thêm 3,125% vào phần thưởng coinbase và người thợ mỏ của U được 93,75% của phần thưởng coinbase tiêu chuẩn.
- Một Uncle bao gồm trong khối B phải có các tính chất sau:
  - Nó phải là con trực tiếp của tổ tiên thế hệ k của B, trong đó 2 <= k <= 7
  - Nó không thể là tổ tiên của B
  - Một Uncle phải có tiêu đề khối hợp lệ, nhưng trước đó không cần xác minh hay thậm chí là một khối hợp lệ
  - Một Uncle phải khác mọi Uncle khác bao gồm trong các khối trước và tất cả những Uncle khác nằm trong cùng một khối (không bao gồm hai lần)

3.3 Sự khác biệt giữa Ethereum và Bitcoin

Về nguồn gốc, Bitcoin được tạo ra như một loại tiền tệ và để lưu trữ giá trị. Còn Ethereum được tạo ra như một nền tảng giao dịch hợp đồng thông minh phân tán. Lưu ý rằng Bitcoin cũng có thể xử lý được hợp đồng thông minh, và Ethereum cũng có thể được sử dụng như một loại tiền tệ. Ngoài ra, giữa Bitcoin và Ethereum còn có những điểm khác biệt cơ bản sau:

Điểm khác biệt	Bitcoin	Ethereum
Nguồn gốc	Được tạo ra như một loại tiền tệ và để lưu trữ giá trị	Được tạo ra như một nền tảng giao dịch hợp đồng thông minh phân tán
Thời gian tạo khối mới	10 phút	14 - 15 giây
Tốc độ giao dịch	Chậm	Nhanh
Phí giao dịch	Bị giới hạn, bị cạnh tranh trực tiếp với nhau để được vào khối của Bitcoin	Được tính dựa trên khối lượng tính toán, băng thông lưu trữ
"Đào mỏ" tập trung	Cho phép	Không cho phép

Bảng 1: Bảng so sánh giữa Bitcoin và Ethereum

3.4 Ứng dụng của Ethereum

3.4.1 Hiện tại

Hệ thống thanh toán

Nếu xét về khía cạnh ứng dụng hệ thống thanh toán thì Ethereum cũng tương tự như Bitcoin. Nếu đồng tiền Bitcoin là một ứng dụng của chuỗi khối Bitcoin thì Ethereum cũng là một ứng dụng của chuỗi khối Ethereum. Ethereum từ khi ra đời đã có nhiều cuộc tranh cãi cho rằng nó có thể sử dụng để lưu trữ giá trị đồng tiền. Hầu hết các thanh toán trong mạng lưới Ethereum được xác nhận bởi các thợ đào mỏ và được ghi chép, lữu trữ vào cuốn sổ cái công khai, như nhau cả.

Đầu tư vàng

Mạng lưới Ethereum đã được các kỹ sư của Digix sử dụng để xây dựng ứng dụng đầu tư vàng. Digix có thể được sử dụng để mua những token vàng bằng tiền mặt hoặc sử dụng đồng tiền Ether. Những token này được mã hóa rất phức tạp và liên kết chặt chẽ với các thợ mỏ vàng tại Singapore. Trường hợp Digix bị phá sản, bạn vẫn thể đổi các token vàng này thành vàng thật mà không cần tới bên trung gian nào như ngân hàng, các nhà môi giới hay tiệm vàng.

Gây quỹ công cộng

Những tổ chức như Indiegogo hay Kickstarter đã sử dụng mạng lưới của Ethereum nhằm xây

dựng và phát triển hệ thống gây quỹ cộng đồng. Việc cần làm chỉ là xác định ý tưởng và mục tiêu cho nguồn quỹ, khi thành công Kickstarter sẽ lấy 5% từ nguồn quỹ, số còn lại sẽ chuyển đến các quỹ starup. Mạng dưới chuỗi khối Ethereum được các nhóm khởi nghiệp sử dụng để đặt mục tiêu cho ngân sách, việc còn lại sẽ được thực hiện bởi các hợp đồng thông minh nếu thành công.

Quản lý tài chính doanh nghiệp

Tháng 5/2016, Quỹ cộng đồng lớn nhất trên thế giới có tên gọi The DAO được thành lập. Bản chất của The DAO là một quỹ đầu tư mạo hiểm dựa trên hệ thống biểu quyết phân cấp nhờ hợp đồng thông minh từ đó đưa ra những quyết định đầu tư hợp lý. Đây được coi là một thí nghiệm mang tính cách mạng cho loài người. Tuy nhiên, dự án này vẫn chưa thể triển khai, nếu The DAO thành công, thế giới sẽ chứng kiến một viễn cảnh khác khi các doanh nghiệp sử dụng mô hình quản lý bằng chuỗi khối, việc này đồng nghĩa sẽ không có chức vụ Chủ tịch Hội đồng quản trị, Tổng giám đốc hay các Phó - Trưởng phòng nữa.

3.4.2 Trong tương lai

Internet của vạn vật

Đây có thể nói là một lý tưởng mà con người luôn muốn hướng tới. Một khi đạt được nó, mọi vật trên thế giới này sẽ được kết nối lại với nhau mà không có sự tương tác giữa con người với máy tính hay con người với con người. Bằng cách sử dụng một thiết bị gọi là Ethereum Computer (https://slock.it/ethereum_computer.html), mọi vật hay tài sản đều được hệ thống hóa quản lý trong một không gian kỹ thuật số.

Một ví dụ cụ thể: Giả sử với một chiếc máy rút tiền ATM khi hết tiền thì sẽ cần sự can thiệp của con người để kiểm tra xem tình trạng máy và quyết định đưa thêm tiền vào. Chưa hết muốn thêm tiền thì con người phải ký hàng loạt các giấy tờ và làm báo cáo có liên quan. Thực sự mất thời gian và rất rườm rà, còn với Ethereum Computer, các máy ATM sẽ tự liên kết với hệ thống kế toán và thêm tiền vào máy mà không cần sự can thiệp của con người, đảm bảo được tính bảo mật, độ tin cậy cao nhất.

Thị trường dự đoán

Gnosis (https://gnosis.pm/) và Augur (https://augur.net/) chính là 2 ứng dụng thị trường dự đoán dựa trên mạng lưới của Ethereum.

Lưu trữ web

Swarm là một trong những dự án của đồng tiền ảo Ethereum được phát triển bởi Viktor Tron. Qua đó Swarm sẽ cung cấp dịch vụ lưu trữ web cho nhiều người. Một trang web phân cấp sẽ được

lưu trữ ở mọi nơi trong cùng một thời điểm. Việc này giúp phát hiện mọi hành vi tấn công DdoS hay ai đó đang tìm cách phá hoại website và ngăn chặn nó không được thực hiện, trừ khi hành vi đó được cấp phép trên toàn bộ hệ thống chuỗi khối. Bên cạnh đó cũng hỗ trợ hay thậm chí ngăn chặn việc kiểm duyệt và đánh sập website bởi các thế lực Chính phủ.

Mạng xã hội

Chắc hẳn bạn không thể không biết tới Mark Zuckerberg người đứng đầu của mạng xã hội Facebook lớn nhất thế giới. Có thể bạn chưa biết Mark Zuckerberg vẫn đọc tin nhắn của bạn hằng ngày, bạn có quan tâm về việc những tấm ảnh mà bạn đã xóa vẫn còn tồn tại trên hệ thống facebook? Facebook hay bất kỳ mạng xã hội nào khác cũng vậy, chúng đều được quản lý bởi một bộ máy trung tâm. Vì thế, mạng xã hội Akasha (http://akasha.world) được ra đời, một ứng dụng của mạng lưới Ethereum. Một khi Akasha hoạt động, bạn sẽ giải đáp được tất cả các câu hỏi liên quan đến tính riêng tư khi bạn tham gia vào mạng xã hội của mình.

Chuyển nguồn năng lượng

Với ứng dụng Ethereum bạn có thể chuyển nguồn năng lượng từ tấm pin mặt trời sang cho nhà bên cạnh một cách hoàn toàn tự động khi đã có đủ năng lượng sử dụng hoặc bạn có thể bán nó nếu không muốn chia sẻ miễn phí. Thật tuyệt vời phải không nào? Ứng dụng TransActive Grid (http://transactivegrid.net/) này sẽ giúp bạn sử dụng pin mặt trời một cách hiệu quả hơn.

Giấy kết hôn – di chúc

Nghe có vẻ xa vời nhưng điều này hoàn toàn có thể được thực hiện với ứng dụng của Ethereum, bạn có thể lưu trữ giấy kết hôn hay một bản di chúc trên hệ thống chuỗi khối. Hợp đồng thông minh sẽ giúp bạn quản lý mọi điều khoản pháp lý có liên quan. Điều này sẽ sớm xuất hiện trong tương lai gần.

Thị trường tài chính – Bầu cử – Bất động sản

Để mọi việc trở nên công bằng hơn, không có thao túng, gian lận, đã đến lúc kết với nó với chuỗi khối. Mạng lưới này sẽ xác định mọi thứ liên quan, tất cả những gì phải làm và quản lý chúng một cách minh bạch nhất.

4 Các chi tiết trong Ethereum [6]

4.1 Tổng quan

Mục tiêu chính là tạo thuận lợi cho các giao dịch giữa các cá nhân đồng ý giao dịch với nhau nhưng họ lại không thể tin tưởng bất kỳ ai trong giao dịch. Điều này có thể do khoảng cách địa lý giữa họ hoặc có thể là sự không tương thích, năng lực yếu kém, sự không chắc chắn, sự bất tiện hoặc gian dối của các hệ thống pháp luật hiện hành. Bằng cách tạo ra một hệ thống giao dịch thông qua một ngôn ngữ phong phú và rõ ràng chúng ta có thể cải thiện những điểm trên.

Giao dịch trong hệ thống này sẽ có một số thuộc tính ẩn. Sự minh bạch giữa các giao dịch có thể nhìn thấy chính xác thông qua các trạng thái của giao dịch. Đó làm nên sự khác biệt với các giao dịch thông thường vì ngôn ngữ tự nhiên khó hiểu và khó diễn tả đầy đủ các thông tin trong giao dịch.

Nói chung, chúng tôi muốn cung cấp một hệ thống sao cho người dùng có thể được đảm bảo rằng bất kể cá nhân, hệ thống hay tổ chức nào họ tương tác với nhau, họ có thể làm như vậy với sự tự tin tuyệt đối vào những kết quả có thể xảy ra và những kết quả đó có thể xảy ra như thế nào.

Giá trị: Để giao dịch được trong hệ thống thì cần một đồng tiền chung để giao dịch. Để giải quyết vấn đề này, Ethereum có một đồng tiền là Ether, được biết đến như là ETH.Nhưng Đồng tiền cơ sở trong hệ thống này là Wei, tất cả các giao dịch đều được tính bằng một số nguyên của đồng Wei. Một Ether được định nghĩa là 10^18 Wei. Ngoài ra còn có những loại tiền khác Ether như:

$$10^{12}Wie = Szabo \tag{1}$$
$$10^{15}Wie = Finney \tag{2}$$
$$10^{18}Wie = Ether \tag{3}$$

4.2 Mô hình chuỗi khối

Ethereum được xem như một máy trạng thái dựa trên các giao dịch. Chúng ta bắt đầu với trạng thái khởi tạo và thực hiện các giao dịch để dần biến đổi nó tới trạng thái kết thúc. Mỗi trạng thái có thể bao gồm một số thông tin như số dư tài khoản, tên tài khoản, dữ liệu liên quan đến thông tin về thế giới vật lý hay bất cứ điều gì mà có thể đại diện bởi một máy tính được chấp nhận. Vì vậy, các giao dịch phải đại diện cho đường vòng cung nối giữa hai trạng thái hợp lệ. Sự hợp lệ trong giao dịch là vô cùng quan trọng. Quá trình chuyển đổi trạng thái hợp lệ là một quá trình chuyển đổi thông qua giao dịch. Như sau:

$$\sigma_{t+1} \equiv \Upsilon(\sigma_t, T) \tag{4}$$

Trong đó Υ là hàm chuyển đổi trạng thái trong Ethereum. Trong Ethereum,Υ cùng với σ

mạnh hơn đáng kể so với bất kỳ hệ thống nào đang có. Υ cho phép có thể tính toán tùy ý, trong khi đó σ cho phép lưu các trạng thái tùy ý giữa các giao dịch.

Các giao dịch được sắp xếp thành các khối; các khối được nối liền với nhau bằng cách sử dụng một băm mật mã. Các khối hoạt động như một nhật ký, ghi lại một loạt các giao dịch cùng với khối trước đó và một định danh cho trạng thái cuối cùng (mặc dù không lưu trữ trạng thái cuối cùng chính nó | sẽ quá lớn). Hệ thống cũng chấm dứt chuỗi giao dịch với phần thưởng cho người đào. Sự khuyến khích này diễn ra như là một chức năng chuyển đổi trạng thái, bổ sung giá trị cho một tài khoản được chỉ định.

Nó được giải thích như sau:

$$\sigma_{t+1} \equiv \Pi(\sigma_t, B) \tag{5}$$

$$B \equiv (..., (T_0, T_1, ...)) \tag{6}$$

$$\Pi(\sigma, B) \equiv \Omega(B, \Upsilon(\Upsilon(\sigma, T_0), T_1)...) \tag{7}$$

Trong đó Ω là chức năng chuyển trạng thái hoàn thiện khối(chức năng thưởng cho bên chỉ định); B là khối này, bao gồm một loạt các giao dịch giữa một số thành phần khác; và Π là cấp của khối trong hàm chuyển trạng thái.

Đây là cơ sở của mô hình BlockChain, một mô hình tạo thành xương sống không chỉ của Ethereum, mà còn cho đến nay tất cả các hệ thống giao dịch dựa trên sự đồng thuận giữa các bên.

4.3 Những quy tắc

Trong bài, chúng tôi sử dụng một số ký hiệu chính thức được tất mọi người công nhận.

Các trạng thái của giao dịch được biểu thị bằng σ (hoặc một biến thể ở đó) và các trạng thái của máy, μ.

Các hàm hoạt động dựa trên các giá trị có cấu trúc cao được biểu thị bằng chữ hoa tiếng Hy Lạp, ví dụ: Υ, Ethereum chức năng chuyển trạng thái.

Đối với hầu hết các chức năng, một chữ cái viết hoa được sử dụng, ví dụ C: hàm về chi phí chung. Đây có thể được ký hiệu để biểu thị các biến thể chuyên biệt, ví dụ: CSSTORE, chức năng chi phí cho hoạt động SSTORE. Đối với các chức năng chuyên biệt và có thể được xác định bên ngoài, tôi có thể định dạng dưới dạng văn bản đánh máy, ví dụ: chức năng băm Keccak-256 (theo mục nhập chiến thắng của cuộc thi SHA-3) được biểu thị là KEC (và thường được gọi là Keccak đơn giản). Ngoài ra KEC512 là đề cập đến chức năng băm Keccak 512.

Giá trị vô hướng và các chuỗi byte có kích thước cố định (hoặc, synony-mously, mảng) được biểu thị bằng một trường hợp thường thấp hơn, ví dụ: n được sử dụng trong tài liệu để biểu thị nonce giao dịch. Những người có một ý nghĩa đặc biệt đặc biệt có thể là tiếng Hy Lạp, ví dụ: δ, số lượng các mục cần thiết trên ngăn xếp cho một hoạt động nhất định.

Trong toàn bộ, chúng ta giả sử các đại lượng vô hướng là các số nguyên dương và do đó thuộc về tập hợp P. Tập các dãy tất cả các byte là B, được định nghĩa chính thức trong Phụ lục B. Nếu

như một chuỗi các trình tự được giới hạn ở những đoạn có độ dài đặc biệt, một subscript, do đó tập của tất cả các chuỗi byte có độ dài 32 được đặt tên B_{32} và tập hợp của tất cả các số nguyên dương nhỏ hơn 2^{256} được đặt tên P_{256}. Điều này được định nghĩa chính thức trong phần tiếp theo.

Dấu ngoặc vuông được sử dụng để lập chỉ mục và tham chiếu các thành phần riêng lẻ hoặc các chuỗi con của các trình tự, ví dụ: $\mu_s[0]$ biểu thị mục đầu tiên trên ngăn xếp của máy. Đối với chuỗi con, các điểm elip được sử dụng để xác định phạm vi dự định, bao gồm các phần tử ở cả hai giới hạn, ví dụ: μ_m $[0 :: 31]$ biểu thị 32 đầu tiên của bộ nhớ máy.

4.4 Khối, trạng thái và các giao dịch

Sau khi giới thiệu các khái niệm cơ bản đằng sau Ethereum, chúng tôi sẽ thảo luận chi tiết hơn về ý nghĩa của giao dịch, khối và trạng thái

4.4.1 Thế Giới Trạng Thái (World State)

Trạng thái, là một ánh xạ giữa các địa chỉ (các định danh 160-bit) và các trạng thái của tài khoản. Mặc dù không được lưu trữ trên chuỗi khối, giả định rằng việc thực hiện sẽ duy trìmô hình này trong một cây Merkle Patricia được sửa đổi (trie). Trie là cơ sở dữ liệu đơn giản duy trì một mô hình của bytearrays; chúng ta đặt tên cơ sở dữ liệu cơ bản là cơ sở dữ liệu trạng thái. Điều này có một số lợi ích; đầu tiên nút gốc của cấu trúc này tùy thuộc mật mã vào tất cả các dữ liệu bên trong và do đó mã băm của nó có thể được sử dụng như một khóa an toàn cho toàn bộ trạng thái hệ thống.

Thứ hai,đó là một cấu trúc dữ liệu không thay đổi, nó cho phép bất kỳ trạng thái trước đó được gọi lại bằng cách đơn giản thay đổi gốc rễ tương ứng. Vì chúng ta lưu trữ tất cả các rễ gốc như vậy trong chuỗi khối, chúng ta có thể trở lại trạng thái cũ. Trạng thái tài khoản bao gồm bốn lĩnh vực sau:

- **nonce**: Là một giá trị vô hướng được tính bằng số lượng các giao dịch được gửi từ địa chỉ này hoặc, trong trường hợp tài khoản có mã liên quan, số lượng hợp đồng tạo ra bởi tài khoản này. Đối với địa chỉ một trong trạng thái σ, điều này sẽ được ký hiệu là $\sigma[a]_n$.
- **balance**:Là một giá trị vô hướng được tính bằng Wei của tài khoảng đó. Nó được ký hiệu là $\sigma[a]_n$.
- **storageRoot**:Một mã băm 256 bit của nút gốc của cây Merkle Patricia mã hóa nội dung lưu trữ của tài khoản (một ánh xạ giữa các giá trị số nguyên 256-bit), được mã hoá thành cây trie. Mã hóa đó ký hiệu là $\sigma[a]_s$.
- **codeHash**: là mã băm của máy ảo EVM của tài khoản đó-đây là đoạn mã được thực hiện khi địa chỉ này nhận được một cuộc gọi tin nhắn; nó là không thay đổi và do đó, không giống như tất cả các lĩnh vực khác, không thể thay đổi sau khi xây dựng. Tất cả các đoạn mã như vậy được chứa trong cơ sở dữ liệu trạng thái dưới các giá trị tương ứng của nó để truy xuất sau đó. Băm này được ký hiệu là $\sigma[a]_c$, và do đó mã có thể được ký hiệu là b, và KEC (b) $= \sigma[a]$.

4.4.2 Địa chỉ (Homestead)

Là một số khối quan trọng mang tính công khai đánh dấu giữa sự chuyển tiếp giữa hai trạng thái, chúng ta ký hiệu nó bằng N_h, được định nghĩa như sau:

$$N_h \equiv 1150000 \tag{8}$$

Giao thức này đã được nâng cấp tại mỗi khối, do đó biểu tượng này xuất hiện trong một số phương trình để giải thích cho sự thay đổi.

4.4.3 Giao dịch (Transaction)

Giao dịch (giao dịch chính thức, T) là một chỉ lệnh được mã hoá bằng ký tự được xây dựng bởi một hệ thống bên ngoài phạm vi của Ethereum. Có hai loại giao dịch: những kết quả trong các cuộc gọi thư và những kết quả tạo ra các tài khoản mới có mã liên quan (được gọi là "hợp đồng tạo ra"). Cả hai loại chỉ định một số trường phổ biến:

- **nonce**: là một giá trị vô hướng bằng số lượng các giao dịch của người gửi. Giá trị đó được gọi là T_n.
- **gasPrice**:một giá trị vô hướng bằng số lượng Wei được thanh toán cho mỗi đơn vị gas cho tất cả chi phí tính toán phát sinh do kết quả của việc thực hiện giao dịch này; chính thức là T_n.
- **gasLimit**: Giá trị vô hướng bằng lượng gas tối đa cần được sử dụng trong quá trình thực hiện giao dịch này. Điều này được thanh toán trước, trước khi bất kỳ tính toán được thực hiện và không thể được tăng lên sau đó; được biểu thị bằng T_g.
- **To**: Địa chỉ 160-bit của người nhận cuộc gọi tin nhắn hoặc, đối với giao dịch tạo hợp đồng. ký hiệu: T_t.
- **Giá Trị(Value)**: Một giá trị vô hướng bằng số lượng Wei được chuyển đến người nhận cuộc gọi của tin nhắn hoặc, trong trường hợp tạo hợp đồng, như một khoản tài trợ cho tài khoản mới được tạo; ký hiệu: T_v.
- **v,r,s**: Giá trị tương ứng với chữ ký của giao dịch và được sử dụng để xác định người gửi giao dịch; chính thức là T_w, T_r và T_s.

4.4.4 Khối (block)

Khối trong Ethereum là bộ sưu tập các mẩu thông tin có liên quan (gọi là tiêu đề khối), H, cùng với thông tin tương ứng với các giao dịch bao gồm, T và một tập hợp các phần đầu khối U khác được biết là có một cha mẹ bằng cha mẹ của cha mẹ của khối hiện tại (các khối như vậy được gọi là ommers[2]). Tiêu đề của khối chứa nhiều mẩu thông tin:

- **parentHash**: Kiểu băm 256-bit Keccak của tiêu đề của khối cha, trong toàn bộ; ký hiệu là H_p.
- **ommerHash**: : băm Keccak 256-bit của phần danh sách ommers của khối này; ký hiệu: H_o.
- **beneficiary**: Dịa chỉ 160-bit mà tất cả các khoản phí thu được từ việc khai thác thành công khối này sẽ được chuyển giao; ký hiệu H_c.
- **stateRoot**: Băm 256 bit Keccak của nút gốc trong cấu trúc cây trie được điền vào trong mỗi giao dịch; ký hiệu H_r.
- **transactionsRoot**: Băm 256 bit Keccak của nút gốc của cấu trúc trie được điền với mỗi giao dịch trong phần danh sách giao dịch của khối; ký hiệu H_t.
- **receiptsRoot**: băm Keccak 256-bit của nút gốc của cấu trúc trie với các biên nhận của mỗi giao dịch trong phần danh sách giao dịch của khối; ký hiệu: H_e.
- **logsBloom**: Bộ lọc Bloom bao gồm thông tin có thể lập chỉ mục (địa chỉ logger và chủ đề đăng nhập) chứa trong mỗi mục nhập nhật ký từ mỗi lần nhận được trong danh sách giao dịch; ký hiệu H_b.
- **Độ khó**: là một giá trị vô hướng chỉ mức độ khó trong tính toán của khối này. Nó được tính từ các độ khó của khối trước đó. Ký hiệu H_d
- **number**: Một giá trị vô hướng bằng số khối tổ tiên. Khối nguồn có một số không; ký hiệu H_i.
- **gasLimit** : một giá trị vô hướng bằng mức giới hạn hiện tại của chi phí gas trên mỗi block; ký hiệu: H_l. gasUsed: một giá trị vô hướng bằng tổng lượng gas được sử dụng trong các giao dịch trong khối này; ký hiệu H_g.
- **gasUsed**: một giá trị vô hướng bằng tổng lượng gas được sử dụng trong các giao dịch trong khối này; ký hiệu H_g
- **timestamp**: Một giá trị vô hướng bằng sản lượng hợp lý của thời gian Unix () tại thời điểm khởi đầu của khối này; ký hiệu H_s
- **extraData**: Một mảng byte tùy ý chứa dữ liệu liên quan đến khối này. Đây phải là 32 byte trở xuống; ký hiệu H_x
- **mixHash**: Một băm 256 bit cải thiện sự kết hợp với nonce rằng một số lượng đầy đủ các tính toán đã được thực hiện trên khối này; ký hiệu H_m
- **nonce**: Một mã băm 64 bit được cải thiện với hixHash. Nó được tính bằng tổng số tính toán trong khối đó; ký hiệu H_n.

4.5 Gas và sự thanh toán

Để phòng tránh vấn đề lạm dụng mạng và lảng tránh những câu hỏi không thể tránh khỏi, bắt nguồn từ Turing hoàn chỉnh, tất cả chương trình tính toán trong Ethereum đều phải chịu phí. Biểu phí được quy định trong đơn vị gas (xem phụ lục G cho các chi phí liên quan đến nhiều sự tính toán). Do đó bất kỳ đoạn nhất định nào của lập trình (điều này bao gồm việc tạo ra hợp đồng, tính toán thực hiện thông báo cuộc gọi, sử dụng và truy cập vào tài khoản lưu trữ và thực hiện các hoạt động trên máy ảo) có một sự đồng thuận về chi phí gas.

Mỗi giao dịch có một số lượng gas cụ thể liên kết với nó: gasLimit. Đây là lượng gas mà hoàn toàn được mua lại từ số dư tài khoản của người gửi. Việc mua bán sẽ xảy ra tuỳ theo gasPrice, cũng được chỉ định trong giao dịch. Các giao dịch được coi là không hợp lệ nếu số dư tài khoản không thể hỗ trợ mua hàng như vậy. Nó được đặt tên là gasLimit bởi bất kỳ gas nào chưa sử dụng tại các kết thúc của các giao dịch đều được hoàn lại (tại cùng một tỷ lệ mua hàng) tài khoản của người gửi. Gas không tồn tại bên ngoài thực hiện một giao dịch. Do đó đối với tài khoản tin cậy có mã liên kết, một lượng giới hạn gas tương đối cao có thể thiết lập và bỏ lại.

Nói chung, Ether được dùng để mua gas mà không được hoàn trả được gửi đến địa chỉ người thụ hưởng, địa chỉ của một tài khoản thường dưới sự kiểm soát của người thợ đào mỏ. Giao dịch viên được tự do chỉ định bất kỳ gasPrice mà họ muốn, tuy nhiên người thợ đào mỏ được tự do bỏ qua các giao dịch như họ chọn. Một mức giá gas cao hơn trên một giao dịch do đó sẽ tiêu tốn Ether của người gửi hơn và cung cấp một giá trị lớn hơn cho người thợ mỏ và do đó sẽ nhiều khả năng được lựa chọn để đưa vào bởi nhiều thợ đào mỏ. Nói chung, thợ đào mỏ sẽ chọn để thông báo mức gas tối thiểu mà họ sẽ thực hiện các giao dịch và giao dịch viên sẽ được miễn phí để chấp nhận những giá này trong việc xác định giá gas nào để giao dịch. Bởi vì có một (trọng số) phân phối tối thiểu giá gas có thể chấp nhận, giao dịch viên nhất thiết phải có một sự đánh đổi để làm cho giữa giảm giá gas và tối đa hóa các cơ hội mà giao dịch của họ sẽ được khai thác một cách kịp thời.

4.6 Thực hiện giao dịch

Việc thực hiện giao dịch là một phần phức tạp nhất của giao thức Ethereum: nó định nghĩa một hàm chuyển đổi trạng thái Υ. Giả định rằng bất kỳ giao dịch nào thực hiện trước tiên sẽ vượt qua các kiểm tra ban đầu về tính hợp lệ bên trong của Ethereum. Bao gồm :

- Giao dịch này là một RLP được hình thành tốt, không có thêm các byte tiếp theo;
- chữ ký giao dịch hợp lệ;
- nonce giao dịch hợp lệ (tương đương với nonce hiện tại của tài khoản người gửi);
- giới hạn gas không nhỏ hơn gas tự nhiên, g0, được sử dụng bởi giao dịch;
- số dư tài khoản người gửi có ít nhất chi phí, v0, bắt buộc phải trả trước.

Chúng ta xem xét hàm Υ, với T là một giao tác và σ trạng thái:

$$\sigma' = \Upsilon(\sigma, T)$$

Như vậy σ' là trạng thái hậu giao dịch. Chúng tôi cũng định nghĩa Υ^g để đánh giá lượng gas được sử dụng trong việc thực hiện một giao dịch và Υ^1 để đánh giá các hạng mục giao dịch được ghi lại, cả hai đều được định nghĩa chính thức sau đó.

4.6.1 Trạng thái con

Trong suốt quá trình thực hiện giao dịch, chúng tôi tích luỹ một số thông tin được thực hiện ngay sau khi giao dịch. Chúng ta gọi trạng thái con giao dịch này và ký hiệu nó là A, là một tuple:

$$A \equiv (A_s, A_1, A_r)$$

Các nội dung tuple bao gồm A_s, bộ tự hủy: một bộ tài khoản sẽ được loại bỏ sau khi hoàn thành giao dịch. A_1 là chuỗi đăng nhập: đây là một loạt các điểm kiểm tra được lưu trữ và có thể lập chỉ mục trong việc thực hiện mã VM cho phép các cuộc gọi theo hợp đồng dễ dàng theo dõi bởi những người xem bên Ethereum (chẳng hạn như các ứng dụng phân quyền trước). Cuối cùng là A_r, số dư hoàn lại, tăng lên thông qua việc sử dụng lệnh SSTORE để thiết lập lại dung lượng lưu trữ hợp đồng bằng 0 từ một số giá trị khác không. Mặc dù không được hoàn trả ngay lập tức nhưng nó được phép bù đắp một phần chi phí thực hiện.

Tóm lại, chúng ta xác định các trạng thái con trống A_0 để không có tự hủy, không có bản ghi và hoàn lại cho Balance một giá trị bằng không:

$$A^0 \equiv (\varnothing, (), 0)$$

4.6.2 Thực hiện

Chúng tôi xác định số gas có sẵn g_0, lượng gas giao dịch này đòi hỏi phải được thanh toán trước khi thực hiện, như sau:

$$g_0 = \sum_{i \in T_i, T_d} \begin{cases} G_{txdatazero} & i = 0 \quad (9a) \\ G_{txdatannonzero} & i \neq 0 \quad (9b) \end{cases}$$

$$+ \begin{cases} G_{txcreate} & T_t = \varnothing \quad (10a) \\ 0 & (10b) \end{cases}$$

$$+ G_{transaction}$$

trong đó T_i; T_d là chuỗi các byte của dữ liệu liên quan đến giao dịch và mã EVM khởi tạo, tùy thuộc vào việc giao dịch là để tạo hợp đồng hoặc gửi tin nhắn. $G_txcreate$ được thêm vào nếu giao dịch là hợp đồng tạo ra, nhưng không phải nếu một kết quả của EVM-mã hoặc trước khi chuyển đổi Homestead.

Chi phí trả trước v_0 được tính như sau:

$$v_0 \equiv T_g T_p + T_v$$

Hiệu lực được xác định như sau:

$$S(T) \neq \varnothing\Lambda$$
$$\sigma[S(T)] \neq \varnothing\Lambda$$
$$T_n = \sigma[S(T)]_n\Lambda$$
$$g_0 \leq T_g\Lambda$$
$$v_0 \leq \sigma[S(T)]_b\Lambda$$
$$T_g \leq B_{Hl} - l(B_R)_u$$

Lưu ý điều kiện cuối cùng; tổng của giới hạn gas của giao dịch, T_g, và gas sử dụng trong khối này trước, được đưa ra bởi $'(B_R)_u$, không được lớn hơn gasLimit của khối,B_{Hl}.
Việc thực hiện một giao dịch hợp lệ bắt đầu với một sự thay đổi không thể huỷ ngang được thực hiện đối với trạng thái: sự nonce của tài khoản của người gửi, S (T), được gia tăng bởi một và số dư được giảm bởi một phần chi phí lên trước, T_gT_p. Gas có sẵn để tính toán tiến trình, g, được định nghĩa là $T_g - g_0$. Việc tính toán, dù là tạo hợp đồng hay một cuộc gọi thư, kết quả là một trạng thái cuối cùng (hợp pháp có thể tương đương với trạng thái hiện tại), sự thay đổi được xác định và không bao giờ hợp lệ: có thể không có giao dịch không hợp lệ từ thời điểm này.
Chúng ta xác định trạng thái trạm kiểm soát σ_0:

$$\sigma_0 = \sigma$$

$$\sigma_0[S(T)]_b = \sigma_0[S(T)]_b - T_gT_p$$
$$\sigma_0[S(T)]_n = \sigma_0[S(T)]_n + 1$$

Đánh giá σ_p từ σ_0 phụ thuộc vào loại giao dịch; tạo hợp đồng hoặc gửi tin nhắn; chúng ta xác định tuple trạng thái tạm thời sau khi thực hiện σ_p, còn lại gas g_0 và trạng thái con A:

$$(\sigma_P, g', A \equiv) \begin{cases} \Lambda(\sigma_0, S(T), T_o, g, T_p, T_v, T_i, 0) & T_t = \varnothing \quad (11a) \\ \Theta_3(\sigma_0, S(T), T_o, T_t, T_t, g, T_p, T_v, T_v, T_d, 0) & (11b) \end{cases}$$

g là lượng gas còn lại sau khi trừ đi số tiền cơ bản cần thiết để trả cho sự tồn tại của giao dịch:

$$g \equiv T_g - g_0$$

và T_o là giao dịch ban đầu, có thể khác với người gửi trong trường hợp một cuộc gọi thư hoặc hợp đồng tạo không trực tiếp gây ra bởi một giao dịch mà đến từ việc thực hiện mã EVM.

Lưu ý chúng ta sử dụng Θ_3 để biểu thị thực tế là chỉ có ba thành phần đầu tiên của giá trị chức năng được lấy; cuối cùng đại diện cho giá trị đầu ra của một cuộc gọi (một mảng byte).
Sau khi cuộc gọi thông báo hoặc hợp đồng được tạo xong, trạng thái sẽ được hoàn thành bằng cách xác định số tiền được hoàn lại, g* từ gas gas còn lại, g_0, cộng thêm khoản trợ cấp từ bộ phận hoàn trả cho người gửi ở mức ban đầu.

$$g* = g' + \min \left\lfloor \frac{T_g - g'}{2} \right\rfloor, A_r$$

Tổng số tiền hoàn lại là gas còn lại g_0 hợp pháp, được thêm vào A_r, với thành phần thứ hai được giới hạn tối đa một nửa (làm tròn) của tổng số tiền được sử dụng $T_g - g_0$.

Các Ether cho gas được trao cho người khai thác mỏ, mà địa chỉ được xác định như là người thụ hưởng của khối hiện tại B. Vì vậy, chúng tôi xác định trước khi trạng thái σ^* về trạng thái tạm thời σ_P:

$$\sigma^* \equiv \sigma_P$$
$$\sigma^*[S(T)]_b \equiv \sigma_P[S(T)]_b + g * T_p$$
$$\sigma^*[m]_b \equiv \sigma_P[m]_b + (T_g - g*)T_p$$
$$m \equiv B_{H_c}$$

Trạng thái cuối cùng, σ_0, đạt được sau khi xóa tất cả các tài khoản xuất hiện trong danh sách xóa:

$$\sigma' \equiv \sigma^*$$
$$\sigma' \equiv \varnothing$$

Và cuối cùng, chúng ta chỉ định Υ^g, tổng lượng gas được sử dụng trong giao dịch này và Υ^1, các bản ghi được tạo ra bởi giao dịch này:

$$\Upsilon^g(\sigma, T) \equiv T_g - g'$$
$$\Upsilon^1(\sigma, T) \equiv A_1$$

4.7 Tạo hợp đồng

Có một số các tham số bên trong được sử dụng khi tạo một tài khoản: người gửi (s), giao dịch ban đầu (o), gas có sẵn (g), giá gas (p), sự đóng góp (v) cùng với mảng byte chiều dài tùy ý, i, mã EVM khởi tạo và cuối cùng là chiều sâu hiện tại của ngăn xếp messagecall / hợp đồng được tạo (e).
Chúng ta xác định chức năng tạo hợp đồng như là hàm Λ, được đánh giá từ các giá trị này, cùng với trạng thái σ đến bộ chứa trạng thái mới, còn lại của gas và tích lũy giao dịch tích luỹ $(\sigma_0; g_0; A)$, như sau:

$$(\sigma', g', A) \equiv \Lambda(\sigma, s, o, g, p, v, i, e)$$

Địa chỉ của tài khoản mới được định nghĩa là 160 bit bên phải nhất của băm Keccak của mã RLP của cấu trúc chứa chỉ người gửi và nonce. Như vậy chúng ta xác định địa chỉ kết quả cho tài khoản mới a:

$$a \equiv \beta_{96...255}(KEC(RLP(s, \sigma[s]_n - 1)))$$

Trong đó KEC là hàm băm Keccak 256-bit, RLP là hàm mã hóa RLP, $B_{a::b}(X)$ đánh giá giá trị nhị phân chứa các bit các chỉ số trong dải [a, b] của dữ liệu nhị phân X và $\sigma[x]$ là trạng thái địa chỉ của x hoặc $\varnothing$ nếu không có. Lưu ý chúng ta sử dụng một ít hơn giá trị nonce của người gửi; chúng tôi khẳng định rằng chúng tôi đã tăng khoản nonce của tài khoản người gửi trước cuộc gọi này và do đó, giá trị được sử dụng là nonce của người gửi vào lúc bắt đầu giao dịch có trách nhiệm hoặc hoạt động của VM.

Nonce của tài khoản ban đầu được định nghĩa là số không, số dư như giá trị được thông qua, lưu trữ như trống rỗng và mã băm như Keccak 256-bit băm của chuỗi rỗng; số dư của người gửi cũng bị giảm đi bởi giá trị được thông qua. Như vậy trạng thái biến đổi trở thành σ^*:

$$\sigma^* \equiv \sigma$$
$$\sigma^*[a] \equiv (0, v + v', TRIE(\varnothing), KEC()$$
$$\sigma^*[s]_b \equiv sigma[s]_b - v$$

Cuối cùng, tài khoản được khởi tạo thông qua việc thực hiện mã EVM khởi tạo i theo mô hình thực hiện. Việc thực hiện mã có thể ảnh hưởng đến một số sự kiện không thuộc về trạng thái thực hiện: bộ nhớ của tài khoản có thể bị thay đổi, có thể tạo thêm tài khoản và có thể thực hiện các cuộc gọi tin nhắn khác. Như vậy, chức năng thực thi mã Θ đánh giá cho một bộ của trạng thái kết quả σ^{**}, lượng gas còn lại là g**, tích lũy trạng thái con A và thân mã của tài khoản o.

$$(\sigma^{**}, g**, A.o) \equiv \Theta(\sigma^{**}, g, I)$$

Trong đó tham số I được định nghĩa:

$$I_a = a$$
$$I_o = o$$
$$I_p = p$$
$$I_d = ()$$
$$I_s = s$$
$$I_v = v$$
$$I_b = i$$
$$I_e = e$$

I_d đánh giá cho một tuple rỗng vì không có dữ liệu đầu vào cho cuộc gọi này. I_H không được điều trị đặc biệt và được xác định từ BlockChain.

Việc thực hiện mã làm cạn kiệt gas và gas không thể xuống dưới mức 0, do đó việc thực hiện có thể thoát ra trước khi mã đã dừng trạng thái một cách tự nhiên. Trong trường hợp ngoại lệ

này (và một vài trường hợp ngoại lệ khác), chúng tôi nói rằng trường hợp ngoại lệ ngoài gas đã xảy ra: Trạng thái được đánh giá được định nghĩa là tập rỗng,∅, và toàn bộ hoạt động tạo không ảnh hưởng đến trạng thái đó, nó sẽ để lại hiệu quả ngay trước khi thử tạo ra.

Nếu mã khởi tạo hoàn thành thành công, chi phí tạo hợp đồng cuối cùng sẽ được thanh toán, chi phí gửi mã c, c, tỷ lệ thuận với kích thước của mã hợp đồng được tạo ra:

$$c \equiv G_{codedeposit} * |o|$$

Nếu không còn đủ gas để trả chi phí này, tức là $g** < c$, thì chúng tôi cũng nói rằng một ngoại lệ ngoài gas.

Khối lượng còn lại sẽ là số không trong bất kỳ điều kiện đặc biệt nào như vậy, tức là nếu việc tạo ra được thực hiện như một giao dịch, thì điều này không ảnh hưởng đến việc thanh toán chi phí bên trong của hợp đồng; nó được trả lại. Tuy nhiên, giá trị của giao dịch không được chuyển đến địa chỉ của hợp đồng bị hủy bỏ.

Nếu ngoại lệ như vậy không xảy ra, thì gas còn lại được hoàn trả lại cho người khởi tạo và trạng thái đã thay đổi hiện tại được phép tồn tại. Như vậy, chính xác chúng ta có thể chỉ định trạng thái, gas và trạng thái con kết quả như sau: $(\sigma_0; g_0; A)$ trong đó:

$$g' \equiv \begin{cases} 0 & \sigma^{**} = \varnothing & (12a) \\ g^{**} & g** < c \cap H_i < N_h & (12b) \\ g^{**} - c & & (12c) \end{cases}$$

$$\sigma' \equiv \begin{cases} \sigma & \sigma^{**} = \varnothing & (13a) \\ \sigma^{**} & g** < c \cap H_i < N_h & (13b) \\ \sigma'[a]_c = KEC(o) & & (13c) \end{cases}$$

Trường hợp ngoại lệ trong việc xác định σ' cho thấy o, chuỗi byte kết quả từ việc thực hiện mã khởi tạo, xác định phần thân mã cuối cùng cho tài khoản mới tạo ra.

Lưu ý rằng ý định từ khối N_H trở đi (Homestead) là kết quả hoặc là một hợp đồng mới được tạo ra thành công với sự đóng góp của nó, hoặc không có hợp đồng mới mà không có sự chuyển giá trị. Trước khi Homestead, nếu không có đủ gas để trả c, một tài khoản tại địa chỉ mới của hợp đồng sẽ được tạo ra, cùng với tất cả các tác dụng phụ của khởi tạo, và giá trị được chuyển, nhưng không có mã hợp đồng được triển khai.

4.7.1 Sự tinh tế

Lưu ý rằng mặc dù mã khởi tạo được thực hiện, địa chỉ mới tạo được tồn tại nhưng không có bên trong phần thân mã. Vì vậy, bất kỳ cuộc gọi tin nhắn nhận được bởi nó trong thời gian này gây ra không có mã được thực hiện. Nếu việc thực hiện khởi tạo kết thúc bằng một chỉ dẫn tự hủy, vấn đề sẽ được giải quyết vì tài khoản sẽ bị xóa trước khi giao dịch hoàn tất. Đối với một mã dừng bình thường, hoặc nếu mã trả về là trống rỗng, thì trạng thái còn lại với một tài khoản vô nghĩa, và số dư còn lại sẽ bị khóa vào tài khoản mãi mãi.

4.8 Cuộc gọi thông điệp

Trong trường hợp thực hiện một cuộc gọi thông điệp, một số thông số yêu cầu: người gửi (s), giao khởi tạo giao dịch (o), người nhận (r), tài khoản mà mã của người đó được thực hiện (c, thường giống người nhận), gas hiện có (g), giá trị (v) và giá gas (p) cùng với một mảng byte dài tùy ý, d, đầu vào dữ liệu của cuộc gọi và cuối cùng là độ sâu hiện tại của cuộc-gọi-thông-báo/sự-tạo-thành-hợp-đồng (essage-call/contract-creation) (e).

Ngoài việc đánh giá một trạng thái mới và trạng thái giao dịch con, những cuộc gọi thông điệp cũng có một thành phần phụ - lượng dữ liệu được biểu thị bằng mảng byte o. Điều này được bỏ qua khi thực hiện giao dịch, tuy nhiên cuộc gọi thông điệp có thể được bắt đầu do việc thực thi mã-VM và trong trường hợp này thông tin này được sử dụng.

$$(\sigma', g', A, o) = \Theta(\sigma, s, o, r, c, g, p, v, \tilde{v}, d, e) \tag{14}$$

Lưu ý rằng đối với cuộc gọi đại diện (DELEGATECALL) chúng ta cần phải phân biệt giữa giá trị mà được chuyển giao, v, với giá trị bề ngoài trong việc thực thi, v. Chúng ta định nghĩa σ_1, trạng thái chuyển tiếp đầu tiên như là trang thái gốc nhưng với giá trị chuyển nhượng từ người gửi đến người nhận:

$$\sigma_1[r]_b \equiv \sigma_1[r]_b + v \wedge \sigma_1[s]_b - v \tag{15}$$

Thông qua công việc hiện tại, giả định rằng nếu $\sigma_1[r]$ không xác định ban đầu, nó sẽ được tạo ra như là một tài khoản không có mã hoặc trạng thái và không cân bằng. Do đó phương trình trước có nghĩa là:

4.9 Mô hình thực thi

Mô hình thực thi quy định cụ thể cách trạng thái hệ thống được thay đổi một loạt các hướng dẫn bytecode đã cho và dãy hữu hạn nhỏ của dữ liệu môi trường. Điều này được xác định thông qua một mô hình chính thức của một máy trạng thái ảo, được gọi là Ethereum Virtual Machine (EVM). Nó là một máy Turing hoàn chỉnh; tiêu chuẩn gần như xuất phát từ thực tế rằng việc tính toán về bản chất bao quanh thông qua một tham số, gas, điều này làm hạn chế tổng số tính toán được thực hiện.

4.9.1 Khái niệm cơ bản

EVM là một kiến trúc dựa trên ngăn xếp đơn giản. Kích thước chữ của máy (và do đó kích thước của ngăn xếp) là 256-bit. Điều này đã được lựa chọn để tạo thuận lợi cho hàm băm Keccak-256 và tính toán đường cong elip. Các mô hình bộ nhớ là một mảng byte địa chỉ từ đơn giản. Các ngăn xếp có kích thước tối đa là 1024. Máy cũng có một mô hình lưu trữ độc lập; điều này cũng tương tự như khái niệm đối với bộ nhớ nhưng thay vì một mảng byte, nó là một mảng chữ có thể được đánh địa chỉ (wordaddressable). Không giống như bộ nhớ không ổn định, lưu trữ là không dễ thay đổi và được duy trì như một phần của trạng thái hệ thống. Tất cả các vị trí trong cả lưu

trữ và bộ nhớ đều được xác định ban đầu là 0. Máy không tuân theo tiêu chuẩn kiến trúc von Neumann. Thay vì lưu trữ mã chương trình trong bộ nhớ thường có thể truy cập hoặc lưu trữ, nó được lưu trữ riêng biệt trong một ROM ảo chỉ thông qua một hướng dẫn chuyên biệt.

Máy có thể được thực thi ngoại lệ trong vài lý do, trong đó có ngăn xếp luồng dưới và hướng dẫn không hợp lệ. Giống như ngoại lệ hết gas (OOG), chúng không để lại sự thay đổi trạng thái nguyên vẹn. Thay vào đó, máy ngừng ngay lập tức và báo cáo vấn đề này với tác nhân thực thi (hoặc bộ vi xử lý giao dịch hoặc môi trường thực thi) cái mà sẽ giải quyết nó một cách riêng biệt.

4.9.2 Tổng quan về phí

Phí (được gọi là gas) được trả dưới ba hoàn cảnh khác nhau, cả ba như điều kiện tiên quyết để thực thi việc giao dịch. Điều đầu tiên và phổ biến nhất là phí bản chất cho việc tính toán giao dịch. Thứ hai, gas có thể được khấu trừ để tạo thành các khoản thanh toán cho cuộc gọi thông điệp phụ thuộc hoặc tạo hợp đồng; mẫu thanh toán này tách việc thanh toán thành CREATE, CALL và CALLCODE. Cuối cùng, gas có thể được trả do sự gia tăng trong việc sử dụng bộ nhớ.

Việc thực thi của một tài khoản, tổng lệ phí sử dụng bộ nhớ phải trả tỷ lệ với nhỏ nhất của 32 byte được yêu cầu mà tất cả các chỉ số bộ nhớ (cho dù là để đọc hoặc viết) được bao gồm trong phạm vi. Điều này được trả trên cơ sở chỉ trong một khoảng thời gian; như vậy, sự chỉ dẫn một khu vực bộ nhớ ít nhất 32 byte lớn hơn bất kỳ bộ nhớ được lập chỉ mục trước đó chắc chắn sẽ gây ra thêm phí sử dụng bộ nhớ. Do lệ phí này nó là địa chưa từng vượt qua giới hạn 32-bit. Điều đó nói rằng, việc triển khai phải có khả năng để quản lý tình huống này. Phí lưu trữ có một chút hành vi sắc thái – để khuyến khích giảm thiểu việc sử dụng dung lượng lưu trữ (tương ứng trực tiếp đến một cơ sở dữ liệu trạng thái lớn trên tất cả các nút), lệ phí thực thi cho một giao dịch được xóa một mục trong việc lưu trữ không chỉ từ bỏ, một sự trả lại đủ khả năng được đưa ra; trên thực tế, việc hoàn phí này là một cách hiệu quả trả lên phía trước kể từ khi sử dụng ban đầu của một vị trí lưu trữ chi phí hơn đáng kể hơn so với sử dụng bình thường.

4.9.3 Môi trường thực thi

Ngoài trạng thái hệ thống σ, và lượng gas g còn lại để tính toán, có một vài phần thông tin quan trọng được sử dụng trong các môi trường thực thi mà các tác nhân thực thi phải cung cấp; chúng được chứa trong:

- I_a, địa chỉ của tài khoản công ty sở hữu các mã số thực thi.
- I_o, địa chỉ người gửi của các giao dịch hình thành việc thực thi này.
- I_p, giá gas trong giao dịch hình thành việc thực thi này.
- I_d, các mảng byte dữ liệu đầu vào cho việc thực thi; nếu tác nhân thực thi là một giao dịch, đây sẽ là dữ liệu giao dịch.
- I_s, địa chỉ của tài khoản gây ra các mã để thực thi; nếu tác nhân thực thi là một giao dịch, đây sẽ người gửi giao dịch.

- I_v, giá trị thông qua tài khoản này như một phần của thủ tục tương tự để thực thi; nếu tác nhân thực thi là một giao dịch, đây sẽ là giá trị giao dịch.
- I_b, mảng byte mã máy để thực thi.
- I_H, tiêu đề khối của các khối hiện tại.
- I_e, độ sâu của cuộc-gọi-thông-điệp hoặc việc tạo-hợp-đồng hiện tại (ví dụ số lượng của CALLs hoặc CREATEs đang được thực thi hiện tại).

4.10 Từ cây khối tới chuỗi khối

Các chuỗi khối hợp tiêu chuẩn là một con đường từ gốc đến lá thông qua toàn bộ cây khối. Để có sự thống nhất về con đường đó, chúng ta xác định con đường có tính toán nhiều nhất được thực hiện theo nó, hoặc, con đường có trọng số lớn nhất. Rõ ràng một yếu tố giúp xác định được con đường có trọng số lớn nhất là số khối của lá, tương đương với số khối trên con đường đó, không tính đến khối gene không bị hủy hoại. Con đường càng dài, thì càng cần nhiều nỗ lực để khai thác mỏ và cần phải được thực hiện để đến được lá.

Một khối tiêu đề bao gồm các độ khó,khối tiêu đề đơn lẻ là đủ để xác nhận tính toán cần thực hiện. Bất kỳ khối nào cũng đóng góp vào tổng số tính toán hoặc tổng số độ khó của một chuỗi.

Do đó chúng ta xác định tổng độ khó của khối B đệ quy như sau:

$$B_t \equiv B'_t + B_d$$

$$B' \equiv P(B_H)$$

Trong đó cho biết khối B, B_t là tổng độ khố B' là khối cha và B_d là độ khó.

4.11 Khối hoàn thiện

Quá trình hoàn thiện khối liên quan đến bốn giai đoạn:

- Xác nhận (hoặc nếu khai thác, xác định) ommers;
- Xác nhận hợp lệ (hoặc, nếu khai thác, xác định) giao dịch;
- Áp dụng phần thưởng;
- Xác minh (hoặc, nếu khai thác, tính một hợp lệ) trạng thái và nonce.

4.11.1 Xác nhận ommers

Việc xác nhận các tiêu đề ommer không có gì hơn bằng cách bằng xác minh rằng mỗi tiêu đề ommer gồm xác định tiêu đề hợp lệ và sự đáp ứng các mối quan hệ của ommer đến khối hiện tại.

4.11.2 Xác nhận giao dịch

Loại gas đã sử dụng phải tương ứng trung thực với các giao dịch được liệt kê: B_{H_g}, tổng lượng gas sử dụng trong khối, phải bằng gas tích lũy được sử dụng theo giao dịch cuối cùng:

4.11.3 Ứng dụng thưởng

Việc áp dụng các phần thưởng vào một khối liên quan đến việc nâng cao sự cân bằng của các tài khoản của địa chỉ thụ hưởng của khối. Hệ thống tăng tài khoản thụ hưởng của khối bằng R_b; cho mỗi ommer, chúng ta nâng số người thụ hưởng của khối thêm $\frac{1}{32}$ phần thưởng của khối và người thụ hưởng ommer sẽ nhận được phần thưởng tùy thuộc vào số khối. Ta định nghĩa hàm Ω:

$$\Omega(B, \sigma) \equiv \sigma' : \sigma = \sigma$$

$$\sigma[B_{H_c}]_b = \sigma[B_{H_c}]_b + (1 + \frac{||B_U||}{32})R_b$$

$$\forall U \in B_U : \sigma'[U_c]_b = \sigma[U_c]_b + (1 + \frac{1}{8}(U_i - B_{H_i}))R_b$$

Nếu có va chạm giữa các địa chỉ thụ hưởng giữa các ommers và khối (tức là hai ommers với cùng một địa chỉ thụ hưởng hoặc một ommer có cùng địa chỉ thụ hưởng như khối hiện tại), bổ sung được áp dụng tích lũy.

Chúng ta định nghĩa khối thưởng như 5 Ether:

$$R_b = 5 * 10^{18}$$

4.11.4 Xác nhận trạng thái và nonce

Bây giờ chúng ta có thể định nghĩa hàm,Γ, mà ánh xạ một khối B đến trạng thái khởi đầu của nó:

$$\Gamma(B) \equiv \begin{cases} \sigma & P(B_H) = \varnothing \quad (16a) \\ \sigma_i : TRIE(L_S(\sigma_i)) = P(B_H)_{H_r} & (16b) \end{cases}$$

Ở đây, $TRIE(L_S(\sigma_i))$ có nghĩa là băm của nút gốc của một trie của trạng thái σ_i; giả định rằng việc triển khai sẽ lưu trữ nó trong cơ sở dữ liệu trạng thái, không đáng kể và hiệu quả vì trie tự nhiên là một cấu trúc dữ liệu không thay đổi.

Và cuối cùng xác định Φ, chức năng chuyển khối, trong đó bản đồ một khối B không hoàn chỉnh cho một khối hoàn chỉnh B_0:

$$\Phi(B) \equiv B' : B' = B^*$$

$$B'_n = n : x \leq \frac{2^{256}}{H_d}$$

$$B'_m = m : (x, m) = PoW(B^*, n, d)$$
$$B^* \equiv B : B^*_r = r(\Pi(\Gamma(B), B))$$

Với d là một mảng dữ liệu

Như được xác định ở phần đầu của công việc hiện tại,Π là chức năng chuyển giao trạng thái, được định nghĩa theo thuật ngữ Ω , hàm quyết toán khối và Υ, chức năng thẩm định giao dịch, bây giờ được xác định rõ ràng.

Như đã trình bày ở trên, $R[n]_\sigma$, $R[n]_l$ và $R[n]_u$ là các trạng thái tương ứng thứ n, nhật ký và gas tích luỹ được sử dụng sau mỗi lần giao dịch ($R[n]_b$, thành phần thứ tư của bộ này, đã được xác định theo các nhật ký). Nguyên đơn được xác định đơn giản như là trạng thái kết quả từ việc áp dụng giao dịch tương ứng đến trạng thái phát sinh từ giao dịch trước đó (hoặc trạng thái ban đầu của khối trong trường hợp giao dịch đầu tiên đó):

$$R[n]_\sigma \begin{cases} \Gamma(B) & n < 0 \quad (17a) \\ \Upsilon(R[n-1]_\sigma, B_T[n]) & (17b) \end{cases}$$

Trong trường hợp $B_R[n]_u$, chúng ta sử dụng cách tiếp cận tương tự để xác định mỗi mục như là gas được sử dụng để đánh giá giao dịch tương ứng được tổng kết với mục trước đó (hoặc không, nếu nó là lần đầu tiên), cho chúng ta một tổng số hoạt động:

$$B_R[n]_u \begin{cases} 0 & n < 0 \quad (18a) \\ \Upsilon^g(R[n-1]_\sigma, B_T[n]) + R[n-1]_u & (18b) \end{cases}$$

Đối với $R[n]_l$, chúng ta sử dụng hàm Υ^l mà chúng ta dễ dàng xác định trong chức năng thực hiện giao dịch.

$$R[n]_l = \Upsilon^l(R[n-1]_\sigma, B_T[n])$$

Cuối cùng, chúng ta định nghĩa Π là trạng thái mới với hàm khuyếch đại khối Ω được áp dụng cho trạng thái kết quả của giao dịch cuối cùng, $l(B_R)_\sigma$:

$$\Pi(\sigma, B) \equiv \Omega(B, l(R)_\sigma)$$

4.12 Sự đồng thuận về việc chấp nhận khối

Trên một mạng ngang hàng, mọi dữ liệu ở các nút cục bộ đều có thể bị thay đổi túy ý bởi người sở hữu nút đó và khi thực hiện nhân bản thì sẽ có các xung đột và ta khó có thể xác định đâu mới là chuỗi khối đúng. Vì vậy, để đảm bảo sự tin cậy của mạng cũng như chuỗi khối, ta phải có các cơ chế để làm sao tất các nút trên mạng đều đồng thuận khối thêm vào chuỗi đó là khối hợp lệ và sẽ phát hiện ra các hành vi phá hoại mạng bằng cascch đưa và các khối giả. Nói về cơ chế đồng thuận này thì hiện nay phổ biến nhất sẽ có hai cớ chế là chứng nhận công việc (Proof Of Work - POW) và chứng nhận cổ phần (Proof Of Stake - POS). Ngoài ra còn mô hình chứng nhận thẩm quyền (Proof Of Authority - POA) nhưng trước hết ta sẽ đi vào giải thích POW và POS.

4.12.1 POW

Ta đã quá quen thuộc với việc các thợ mỏ mua các cỗ máy đắt tiền (gồm nhiều card đồ họa) để về đào coin. Việc này chúng tỏ họ đang tham gia vào một mạng lưới tiền ảo sử dụng mô hình POW. POW là một trong các cách để xác định sự đồng thuận của cộng đồng. Ở mô hình hay giải thuật này, để thêm mới một khối và blockchain đồi hỏi phải thực hiện các hàm tính toán rất phức tạp để tạo nên một giá trị băm hợp lệ (khó để tạo ra nhưng rất dễ dàng để xác định nó hợp lệ). Việc giải mã này ngoài việc cần những cấu hình mạnh còn tiêu tốn rất rất nhiều điện năng và gây nguy hại đến môi trường. Cộng đồng sẽ công nhận khối anh tạo ra dựa vào lượng công việc anh đã thực hiện được. Một yếu tố đặc trưng của mô hình này đó là sự xuất hiện của các khối dư thừa (orphan block) do có nhiều người tham gia cùng tạo nên một khối nên sẽ có trường hợp hai người tạo ra hai khối đều hợp lệ nhưng ta chỉ có thể chọn khối đến trước và khối kia (tốn rất nhiều năng lượng tạo ra) bị lãng phí và bỏ đi. Những đặc trưng kể trên phần nào nói lên nhược điểm của giải thuật mà phần lớn các đồng tiền ảo đang sử dụng

Khi độ khó để tạo ra một khối càng ngày càng tăng lên, việc đào ra một khối của các thợ mỏ ngày càng thấp và họ bắt đầu thua lỗ,một số sẽ quyết định từ bỏ hoặc tham gia vào các bể đào (mining pool). Việc này tạo ra các cỗ máy tập trung (trái với các tính chất phân tán mà mô hình mong muốn). Và còn một vấn đề hệ quả nữa là tấn công 51

Nhưng nó phổ biến có nguyên nhân của nó. Giá trị sử dụng của nó là khi một đồng tiền ảo mới được phát hành, lượng tiền chưa được nhiều. Hầu hết các đồng tiền ảo sẽ chọn mô hình này để gia tăng lượng tiền mà vẫn kiểm soát được phần nào lạm phát của nó (điều mà ETH đã làm), tiền chỉ được tạo ra không hề dễ dàng.

4.12.2 POS

Khi mà POW bộc lộ rõ các yếu điểm thì là lúc các mô hình và các giải thuật khác được đề xuất, trong đó có POS. Thực tế thì POS đã được áp dụng ở một số đồng tiền ảo, tiên phong trong đó là PeerCoin và Ethereum sẽ là cái tên tiếp theo áp dụng mô hình này. Mô hình POS thay vì công nhận công việc của anh bằng sức lực anh bỏ ra (chi cho phần cứng và năng lượng) thì lại công nhận bằng nó bằng cổ phần hay số tiền mà anh đặt cọc vào mỗi khối người đó sinh ra. Và giờ đây, mỗi khối không phải là cuộc chạy đua xem ai giải mã được chính xác và nhanh nhất (có thể gây lãng phí với các khối thừa) mà việc tạo khối được chỉ định cho người nào góp cổ phần nhiều nhất vì hệ thống tự động hiểu rằng nếu anh góp vào nhiều tiền như vậy thì anh cũng sẽ có đủ khả năng tính toán để tạo ra khối mới.Và một điều quan trọng nữa là sẽ không còn tồn tại việc thưởng ETH theo mỗi khối đào được mà chỉ trả cho người tạo khối một khoản tiền vì đã thực hiện giao dịch (transaction fee).

Việc sử dụng POS sẽ dẫn đến các lợi ích như sau. Tiết kiệm năng lượng, giúp mạng lưới phân tán hơn, giúp đảm bảo lợi ích người đào.

4.12.3 POA

POA khá khác so với hai mô hình đồng thuận trước, trong khi hai mô hình trên vẫn giữ tính ẩn danh của tất cả những người tham gia và sự tin tưởng khó có thể đạt tới sự chắc chắn (tạm tin tưởng vào lượng công việc cũng như cổ phần) thì thay vào đó ta sẽ có các cơ quan được chứng thực một cách hợp pháp và công khai tham gia vào mạng như các thành viên kiểm chứng. Những cơ quan này sẽ sở hữu các tài khoản và các nút được coi là có thẩm quyền, các thành viên trong mạng có thể tin tưởng hoàn toàn họ và họ sẽ đóng vai trò chính trong việc kiểm tra các giao dịch và các khối có hợp lệ hay không. Mạng theo mô hình này sẽ có một số lợi thế vượt trội so với hai mô hình kia như thời gian tạo khối tương đối ngắn, không lãng phí vì không cần phải đào, giúp cải thiện thời gian triển khai, kiểm thử và ngăn chặn các cuộc tấn công spam. Ethereum đã triển khai một mạng thử nghiệm triển khai mô hình này có tên là Kovan, bạn có thể xem chi tiết hơn tại https://github.com/kovan-testnet/proposal.

4.13 Máy ảo Ethereum (Ethereum Virtual Machine)

Máy ảo Ethereum (EVM) là một máy chủ toàn cầu mà mọi người có thể sử dụng, với một chi phí nhỏ, có thể trả bằng ether

4.13.1 Mạng ngân hàng trung tâm

Chúng ta đang chi rất nhiều tiền cho việc xây dựng, vận hành và bảo trì các hệ thống thanh toán tập trung. Mỗi ngân hàng xây dựng riêng cho mình một hệ thống và việc thanh toán liên ngân hàng sẽ đòi hỏi thêm một khoản phí nữa, chưa kể đến các chi phí để đảm bảo an toàn, an ninh cũng như tính tin cậy cho cả hệ thống. Những chi phí ấy không hề rẻ dẫn đến việc bạn phải chi trả nhiều hơn cho các giao dịch của mình cũng như các giao dịch sẽ bị chậm đi đáng kể khi phải liên kết niều dịch vụ.

4.13.2 Máy ảo

Trong ngữ cảnh của Ethereum, đó là một máy tính toàn cầu bao gồm rất nhiều nút cấu thành và chính các nút đó cũng là các máy tính. Nói chung, máy ảo là mô phỏng một hệ thống máy tính này bằng một hệ thống máy tính khác. Việc mô phỏng này trên EVM sử dụng cả phần cứng và phần mềm, mỗi nút trong mạng có thể chạy bất kì hệ điều hành nào.

4.13.3 Vai trò của EVM

EVM tạo ra môi trường để chạy các chương trình bất kì (được gọi là các hợp đồng thông minh) được viết bằng ngôn ngữ Solidity. Những chương trình này là xác định đầy đủ và đảm bảo là được thực hiện nếu bạn trả đủ chi phí co giao dịch. Các chương trình viết bằng Solidity có khả năng thực hiện tất cả các nhiệm vụ có thể thực hiện được bằng máy tính. Khi một hợp đồng được triển khai bằng việc tải lên từ một nút của mạng, các bản sao của hợp đồng sẽ được phát tán ra

các nút khác. Khi có yêu cầu chạy hợp đồng, các nút trong mạng sẽ chạy một các độc lớp với cùng một mã của hợp đồng. Tuy nó cho thấy sự song song hóa rất cao, nhưng nó cũng mang lại sự dư thừa không hề nhỏ.

4.14 Thực hiện hợp đồng

Có một số mô hình hợp đồng kĩ thuật cho phép thực hiện các việc cụ thể. Trong đó, có hai loại: nguồn cấp dữ liệu và số ngẫu nhiên

4.14.1 Nguồn cấp dữ liệu

Hợp đồng cung cấp dữ liệu là hợp đồng cung cấp một dịch vụ: nó cho phép truy cập thông tin từ thế giới bên ngoài trong Ethereum. Sự chính xác và kịp thời của thông tin này không được đảm bảo và đây là nhiệm vụ của một hợp đồng phụ. Hợp đồng phụ này sẽ sử dụng nguồn cấp dữ liệu để xác định mức độ tin tưởng có thể được đặt trong bất kỳ một nguồn cấp dữ liệu nào. Mô hình tổng quát bao gồm một hợp đồng duy nhất trong Ethereum mà khi nhận được cuộc gọi tin nhắn, trả lời với một số thông tin kịp thời liên quan đến một hiện tượng bên ngoài. Ví dụ như nhiệt độ của thành phố New York. Điều này sẽ thực hiện như một hợp đồng mà trả lại giá trị của một số điểm nhận ra trong lưu trữ. Dĩ nhiên điểm này trong lưu trữ phải được duy trì với nhiệt độ chính xác như vậy, và do đó phần thứ hai của mô hình sẽ cho máy chủ chạy một nút Ethereum, và ngay lập tức phát hiện ra một block mới, gửi đến hợp đồng, cập nhật giá trị trong lưu trữ. Mã của hợp đồng sẽ chấp nhận các cập nhật như vậy từ nhận dạng trên máy chủ.

4.14.2 Số ngẫu nhiên

Cung cấp số ngẫu nhiên trong một hệ thống xác định một cách tự nhiên và không phải nhiệm vụ. Tuy nhiên, chúng ta có thể lấy gần đúng với giả ngẫu nhiên số bằng cách sử dụng dữ liệu nói chung, không thể biết được tại thời điểm giao dịch. Dữ liệu đó có thể bao gồm khối băm, dấu thời gian của khối và lợi ích của khối địa chỉ CIAR. Để các "thợ đào độc hại" khó kiểm soát các giá trị ta nên sử dụng BLOCKHASH operation để sử dụng băm của 256 khối trước như những con số giả ngẫu nhiên. Đối với một loạt các con số như vậy,một giải pháp tầm thường sẽ được thêm một số lượng không đổi và băm kết quả.

4.15 Các thành phần tiếp theo

Các ứng dụng tại máy chủ hiện nay thường sẽ thực hiện ba việc chính sau: tính toán để đưa ra câu trả lời, lưu trữ dữ liệu phản hồi lại tức thì các cử chỉ của người dùng (ứng dụng thời gian thực). Xét lại nền tảng Ethereum chúng ta thấy đã có nền tẳng tính toán là máy ảo Ethereum nhưng hai thành phần còn lại vẫn chưa hỗ trợ. Các nhà lập trình của Ethereum đã thấy điều đó và đã và đang phát triển hai thành phần đó trong các phiên bản tiếp theo của Ethereum. Từ đó ta có các thành phần như sau:

- EVM: Máy trạng thái phi tập trung
- Swarm: Lưu trữ phi tập trung
- Whisper: Gửi tin nhắn phi tập trung

4.15.1 Whisper

Whisper là một hệ thống nhắn tin phi tập trung, là một phần của giao thức Ethreum và sẽ sẵn có cho các ứng dụng web sử dụng EVM như là backend. Không giống với cách thông thường là mọi thay đổi đều được lưu ở chuỗi khối (tuy không thể thay đổi nhưng đắt đỏ và tốn nhiều thời gian), dữ liệu của mỗi tin nhắn sẽ được truyền trực tiếp giữa các hợp đồng thông minh với nhau.

4.15.2 Swarm

Swarm là giao thức dành cho các dữ liệu được đánh địa chỉ. Nó hoạt động với dữ liệu bất biến, phân mảnh nó (sharding) và lưu trữ nó trong mạng phân tán và có thể dễ dàng gọi nó nếu ứng dụng cần. Mục tiêu của Swarm là có thể tìm các phiên bản khác nhau của dữ liệu của một tệp ở cùng một địa chỉ, làm theo hệ phân cấp như các URL hiện nay, tức là có cấu trúc thư mục.

Chính Swarm có thể hiện thực hóa việc đưa một ứng dụng web đơn giản triển khai hoàn toàn trên nền tảng phi tập truung với các tệp HTML, CSS, JS được lưu lại và trả về nếu có yêu cầu từ người dùng, nó sẽ thực sự khác biệt khi ứng dụng web của bạn vươn tới một thứ được gọi là trực tuyến 100% thực sự chứ không phải là 99.99% như hiện tại. Tất nhiên chặng đường vẫn còn xa nhưng thực sự đáng kỳ vọng cho một ứng dụng web như vậy.

4.16 Lộ trình của Ethereum

Mặc dù việc phát triển phần mềm có thể rất khó dự đoán, các nhà phát triển của Ethereum đã có các dự đoán về các mốc thời gian khá rõ ràng. Cụ thể sẽ có các giai đoạn sau.

4.16.1 Frontier Release (2015)

Frontier đã hoàn thành một số mục tiêu đúng hạn. Trong giai đoạn này việc giao tiếp chủ yếu được thực hiện trên các dòng lệnh. Các ưu tiên trong giai đoạn này gồm có:

- Đảm bảo hệ thống mining chạy được
- Hợp pháp hóa đồng ETH như là một loại tiền ảo chính thống
- Phát hành một môi trường để thử nghiệm dapp
- Tạo ra các công cụ để cung cấp ether miễn phí cho các mạng thử nghiệm
- Cho phép các nhà phát triển tải lên và thự thi các hợp đồng thông minh

4.16.2 Homestead Release (2016)

Giai đoạn này cung cấp thêm một công cụ hữu ích hơn cho việc giao tiếp với Ethereum bằng Mist Browser. Các đặc điểm chính của giai đoạn này bao gồm:

- Việc đào Ether đã được trả công 100% giá trị như dự định
- Không có sự ngắt quãng trên toàn mạng
- Gần như bước ra khỏi quá trình thử nghiệm
- Có thêm các tài liệu cho việc sử dụng dòng lệnh và Mist

4.16.3 Metropolis (2017)

Đây là pha thứ hai trong việc phát triển giao thức Ethereum. Bản phát hành này sẽ mở ra việc phát hành chính thức cho Mist với đầy đủ các tính năng. Tiếp theo đó nó sẽ được hỗ trợ bởi các phần mềm bên thứ ba một cách mạnh mẽ. Hơn nữa, Swarm và Whisper sẽ được đưa vào vận hành.

4.16.4 Serenity (2018)

Đây sẽ là giai đoạn chuyển mô hình từ POW to POS với thuật toán có tên gọi Casper. Việc chuyển dịch này là cần thiết với các ưu diểm được nêu từ trước. Quá trình chuyển dịch này sẽ được thực hiện từ từ chứ không chuyển hẳn sang mô hình mới trong một lần. Việc chuyển dịch này sẽ rất cần sự tính toán cũng như chứng minh tính đúng đắn của nó.

5 Ứng dụng bầu cử

Đây là một ứng dụng rất đơn giản để giới thiệu về một trong các ứng dụng quan trọng của nền tảng Ethereum - ứng dụng phi tập trung (decenterlized app). Mọi dữ liệu đều bị ẩn danh và không thể sửa đổi nhưng có một khuyết điểm là để sử dụng nó người dùng cần cài đặt một chương trình để có thể tham gia vào mạng Ethereum, điều này khá khó và phức tạp cho một người dùng cuối. Vì vậy mà đã có rất nhiều ứng dụng hướng việc sử dụng các ứng dụng phi tập trung như một ứng dụng web thông thường, ví dụ như kyber.network. Ứng dụng đơn giản này được viết ra để mô phỏng lại việc đó cũng như hướng dẫn bạn có thể làm được một ứng dụng như thế. Ứng dụng mô tả một cuộc bầu cử đơn giản, những đặc tính của ứng dụng phi tập trung khá phù hợp cho công việc này - ẩn danh và không thể thay đổi.

5.1 Mô tả về yêu cầu ứng dụng

Ứng dụng cho phép người dùng thông thường đăng ký một tài khoản Ethereum mới hoặc nhập lại tài khoản đã có sẵn trên mạng thử nghiệm (ứng dụng này sử dụng mạng thử nghiệm của Ethereum có tên là Rinkeby[7]) mà không cần cài đặt thêm chương trình ngoài, có thể dùng nó như một ứng dụng web.

Trước tiên nên nói về lí do chọn mạng thử nghiệm Rinkeby mà không phải mạng khác (Ethereum còn hai mạng thử nghiệm khác là Ropsten và Kovan tính đến thời điểm viết bài viết này). Lí do đầu tiên chắc chắn là từ sự đơn giản trong việc cài đặt với chỉ một vài dòng lệnh trên Linux, thứ hai là đây là mạng trẻ nhất nên số khối cũng ít nhất nên chúng ta sẽ không phải đồng bộ quá nhiều khối bên cạnh việc chúng ta có thể xin ether miễn phí khá đơn giản và thuận tiện. Những lí do được nêu ra ở trên khá là chủ quan và thiên về việc phát triển ứng dụng, lí do khách quan để mạng thử ngiệm thứ ba được ra đời bắt nguồn từ các yếu điểm của hai mạng thử nghiệm trước đó. Mạng Ropsten hoạt động rất tốt trong thời gian đầu nhưng vì hàm băm dùng cho quá trình đào ở mạng này khá yếu (để các máy tính cá nhân có thể đào được) nên dễ dàng bị những kẻ phá hoại tấn công spam, từ đó mạng Kovan ra đời khắc phục hoàn toàn điểm yếu đó - không còn việc đào ether nữa và các giao dịch được các bên tin cậy xác nhận. Mạng Kovan này hoạt động rất hiệu quả nhưng lại nảy sinh vấn đề với Parity (chương trình cần cài đặt để tham gia mạng này), chỉ những người dùng Parity mới có thể tham gia vào mạng này mà không thể dùng chương tình tùy chọn khác và lí do đó đã thúc đẩy Rikeby ra đời hỗ trợ người dùng có thể dùng các chương trình khác nhau để kết nối với mạng. Vì những ưu thế đó, chúng ta sẽ sử dụng mạng thử nghiệm Rinkeby. Để có thể bầu chọn trong ứng dụng, người sử dụng sẽ cần có ether trong tài khoản bằng cách xin cấp miễn phí tại https://faucet.rinkeby.io. Khi đã ether trong tài khoản của mình, người dùng có thể sử dụng chức năng bầu cử và chỉ được bầu cử một lần duy nhất với mỗi tài khoản Ethereum. Do sử dụng mạng thử nghiệm nên tốc độ phản hồi khá là chậm nên có thể gây phiền toái đôi chút nhưng đủ với một ứng dụng thử nghiệm.

5.2 Tạo lập môi trường giao tiếp với mạng Ethereum

Phần hướng dẫn sau đây là dành cho môi trường phát triển trên Linux (cụ thể hơn là Ubuntu), các môi trường khác có thể thực hiện theo các bước tương tự với một số thay đổi. Để có thể triển khai ứng dụng, cần có một chương trình chạy trên máy chủ giao tiếp với mạng thử nghiệm Rinkeby có tên là Go Ethereum (Geth). Để cài đặt ta thực hiện:

```
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install ethereum
```

Sau khi cài đặt xong để chạy Geth ta sẽ dùng câu lệnh:

```
geth --rinkeby --rpc --rpcapi db,eth,net,web3,personal
```

Ta sẽ thấy một số tham số, –rinkeby để chỉ định mạng thử nghiệm mà chúng ta sử dụng, –rpcapi chỉ định các giao diện lập trình ứng dụng mà ta có thể sử dụng để giao tiếp với chương

trình Geth này. Sau khi thực hiện câu lệnh, bạn sẽ phải đợi một khoảng thời gian đáng kể để chương trình tải về chuỗi khối, bạn phải lưu ý rằng chương trình phải có chuỗi khối mới nhất thì mới có thể thực hiện giao dịch nhưng vẫn có thể nhận ether một cách bình thường hay nói cách khác là có thể nhận giao dịch vào mà không thể thực hiện giao dịch ra.

Bạn cũng có thể dùng luôn bản điều khiển của Geth để kết nối và thực hiện một vài yêu cầu như tạo tài khoản, lấy số dư bằng câu lệnh:

```
geth --datadir=$HOME/.rinkeby attach ipc:$HOME/.rinkeby/geth.ipc console
```

Vậy là ta đã có một nút trên mạng Ethereum và có thể sử dụng web3[8] - một thư viện cung cấp các giao diện lập trình trên mạng Ethereum để thực hiện các yêu cầu theo mong muốn của ứng dụng bầu cử. Việc sử dụng thư viện trên nhằm mục đích giúp ứng dụng web có thể sử dụng các câu lệnh được hỗ trợ trên nút ethereum thông qua một giao diện mà ứng dụng có thể sử dụng được.

5.3 Xây dựng ứng dụng

Sau khi đã thiết lập nút Ethereum và thư viện giao tiếp, ta sẽ tiếp tục với phần vệc xây dựng phần chính (nội dung) cho ứng dụng. Chúng ta sẽ phải xây dựng ứng dụng với ba phần: phần hợp đồng thông minh, phần backend và phần frontend. Những ngôn ngữ ta cần sử dụng là Solidity cho phần hợp đồng thông minh và Javascript cho phần backend và frontend. Phía backend sẽ sử dụng nodejs nên trước khi tiếp tục bạn cần cài đặt nodejs, quá trình cài đặt khá đơn giản và bạn hoàn toàn có thể tự hooàn thành được.

5.3.1 Hợp đồng thông minh

Hợp đồng thông minh của ứng dụng bầu cử này sẽ được viết bằng ngôn ngữ Solidity. Vì chương trình đơn giản nên hợp đồng sẽ khá ngắn, cú pháp của Solidity khá giống với javascript nên tương đối dễ hiểu. Sau khi hoàn thành mã nguồn của hợp đồng thông minh, ta sẽ cần biên dịch mã nguồn đó trước khi có thể triển khai trên máy ảo Ethereum, mỗi lần thay đổi mã nguồn bạn sẽ phải làm lặp lại các bước trên khá là phức tạp và phiền toái, nên ứng dụng này sẽ sử dụng khung làm việc Truffle để hoàn thành các công việc trên một cách tự động.

Một số điều cần lưu ý khi triển khai hợp đồng trên máy ảo Ethereum là bạn cần một tài khoản và ether để triển khai hợp đồng. Để thực hiện các giao dịch cũng như chi tiêu đồng ether này, bạn phải mở khóa tài khoản đó bằng cách chạy lại Geth và thêm tham số –unlock= cộng với giá trị theo sau là địa chỉ tài khoản của bạn, ví dụ –unlock="0x176916e03344eef8B7c85e18D488d71437966333", sau đó chương trình sẽ yêu cầu bạn nhập mật khẩu để xác nhận bạn sở hữu tài khoản đó. Tiếp theo ta cần chỉnh sửa tệp truffle.js để được như sau:

```
module.exports = {
  networks: {
    rinkeby: {
      host: "localhost", // Connect to geth on the specified
      port: 8545,
```

```
      from: "0x176916e03344eef8B7c85e18D488d71437966333",
      network_id: 4,
      gas: 4000000 // Gas limit used for deploys
    }
  }
};
```

Như chứng ta thấy file này sẽ chứa địa chỉ ip, port của nút Ethereum , địa chỉ sẽ chi ether để triển khai, network_id chỉ định mạng sẽ được triển khai, id của mạng Rinkeby là 4, gas chỉ mức bạn chi để triển khai hợp đồng (nếu quá nhỏ sẽ không thể triển khai). Sau khi triển khai xong ta sẽ có tệp Voting.json, tệp này sẽ được chứa các cấu hình để có thể gọi các phương thức của hợp đồng. Cấu trúc của tệp json khá đơn giản gồm mã nguồn đã biên dịch của hợp đồng, địa chỉ hợp đồng trên mạng và một số trường khác bạn có thể tìm hiểu thêm.

5.3.2 Backend

Phía backend sẽ được viết bằng nodejs để đảm bảo tương thích tốt nhất với web3 (đã có các bản phân phối cho các ngôn ngữ khác). Phiên bản web3 sử dụng trong ứng dụng này là phiên bản beta nhưng có nhiều tính năng mới và tương thích tôt hơn, tài liệu sử dụng cụ thể tại https://web3js.readthedocs.io/en/1.0. Việc tạo tài khoản khá đơn giản bằng việc gọi một phương thức của web3, ta chỉ cần chú ý rằng kết quả trả về sẽ gồm một khóa công khai (địa chỉ của tài khoản) và một khóa bí mật, muốn lưu lại khóa bí mật này ta cần mã hóa nó với mật khẩu do người dùng nhập vào, web3 cũng hỗ trợ việc này với phương thức mã hóa AES với độ dài khóa 128 bit. Một điều cần lưu ý dù nó đã được mã hóa với độ dài khóa đủ lớn nhưng cũng phải loại tránh các trường hợp bị lộ thông tin này do mật khẩu của người dùng thường nằm trong tập từ điển nhỏ hơn (để dễ nhớ) nên khả năng bị tấn congo vét cạn vẫn có thể xảy ra. Điều đó cũng khuyên bạn rằng không nên để quá nhiều ether vào trong một tài khoản do dùng một khóa càng nhiều lần thì càng gia tăng khả năng bị lộ khóa.

Vì là ứng dụng thử nghiệm nên ta sẽ chỉ đơn giản định danh mỗi người dùng bởi một token (sẽ được tạo tự động trong lần đầu tiên truy cập), và mỗi tài khoản người dùng có thể tạo nhiều tài khoản Ethereum, mỗi tài khoản này sẽ có thể tải về khóa bí mật của họ (đã được mã hóa trước đó) để sử dụng lại về sau. Điều này cũng được áp dụng trong kyber.network bản thử nghiệm và ví Mist (thực chất Mist được phát triển như là một trình duyệt cho các ứng dụng phi tập trung) nên cũng không quá tồi nếu ta dùng giải pháp này thay vì bắt người dùng đăng ký một tài khoản để lưu trữ các tài khoản Ethereum khác.

Vấn đề chính cần giải quyết là việc mở khóa tài khoản để có thể sử dụng ether trong quá trình bầu cử (muốn thực hiện giao dịch ra cần có khóa bí mật để ký). Về mặt tài liệu, web3 đã cung cấp phương thức mở khóa tài khoản bằng địa chỉ và mật khẩu trong một khoảng thời gian nhất định. Nhưng chú ý rằng bạn không thể dùng phương thức đó nếu bạn không thực hiện bước lưu khóa bí mật đã mã hóa tại thư mục $HOME/.ethereum/rinkeby/keystore. Thông thường khi bạn tạo tài khoản, tệp lưu khóa bí mật này sẽ được tạo tự động nhưng nếu tạo với web3 thì không, vì vậy bạn phải tạo thủ công bằng nghiệp vụ bên ứng dụng web mỗi khi có yêu cầu đăng ký tài khoản Ethereum. Hãy để thời gian mở khóa một cách hợp lý để cân bằng giữa tính tiện dụng và

bảo mật cho ứng dụng, không nên để quá dài.

Việc cần làm tiếp theo là việc chuyển các yêu cầu của người dùng thành các yêu cầu gửi tới hợp đồng để tác động lên chuỗi khối, chú ý rằng đây là mạng thử nghiệm nên nó khá chậm và sẽ mất một khoảng thời gian đáng kể cho mỗi giao dịch. Để chạy chương tình phía backend bạn cần vào thư mục chứa mã nguồn của phần này và chạy câu lệnh, ứng dụng sẽ chạy mặc định tại cổng 3333.

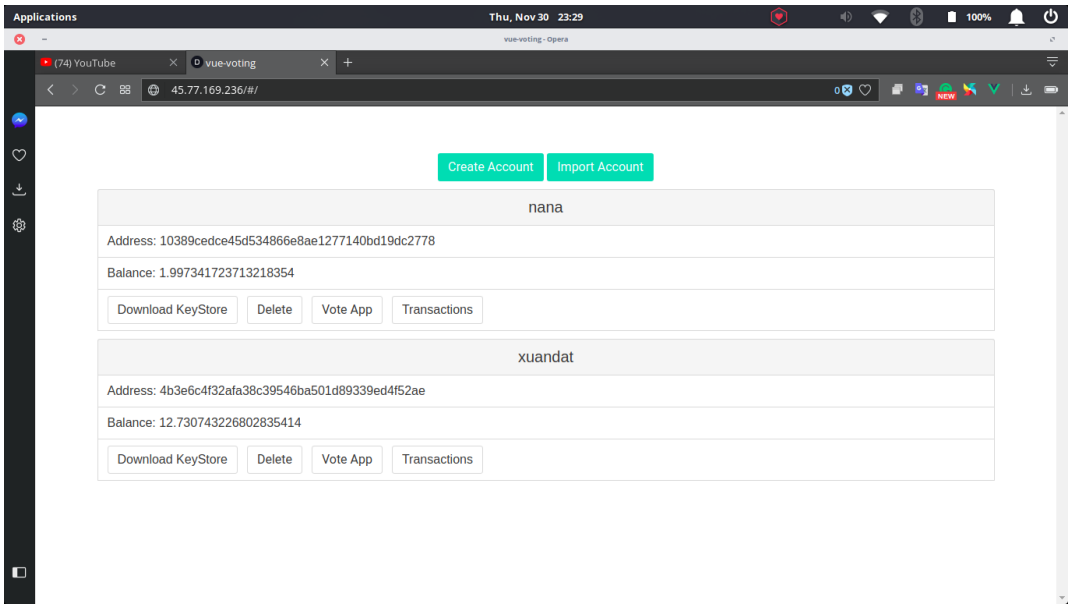
```
npm run server:dev
```

5.3.3 Frontend

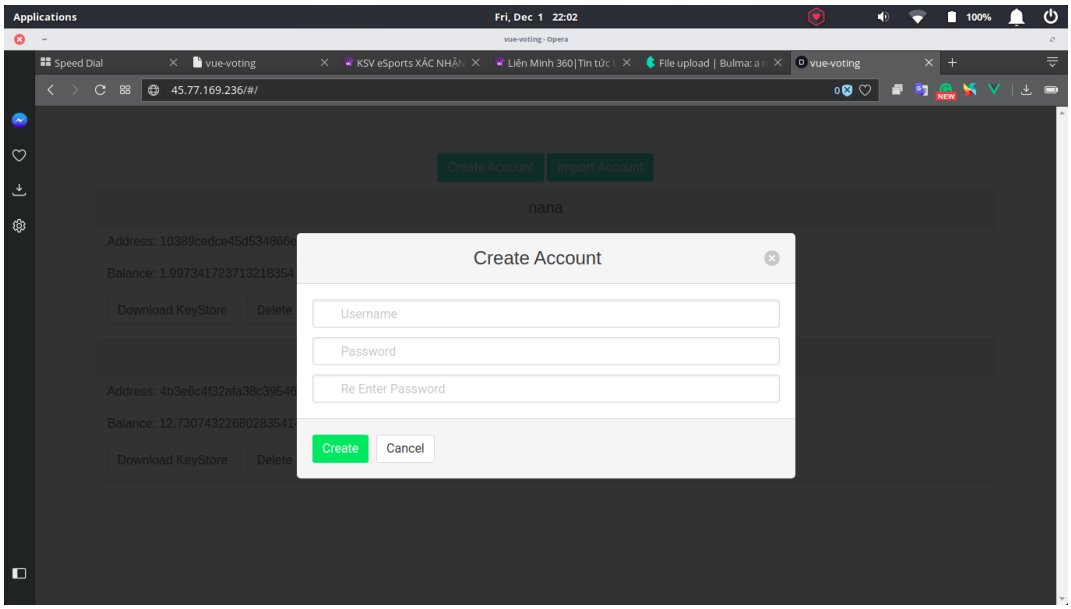
Phía frontend được viết bằng khung làm việc VueJS[9], được sử dụng để xây dựng ứng dụng một trang (Single Page Application) một cách nhanh chóng. Để chạy chương trình phía frontend bạn cần chạy câu lệnh:

```
npm run dev
```

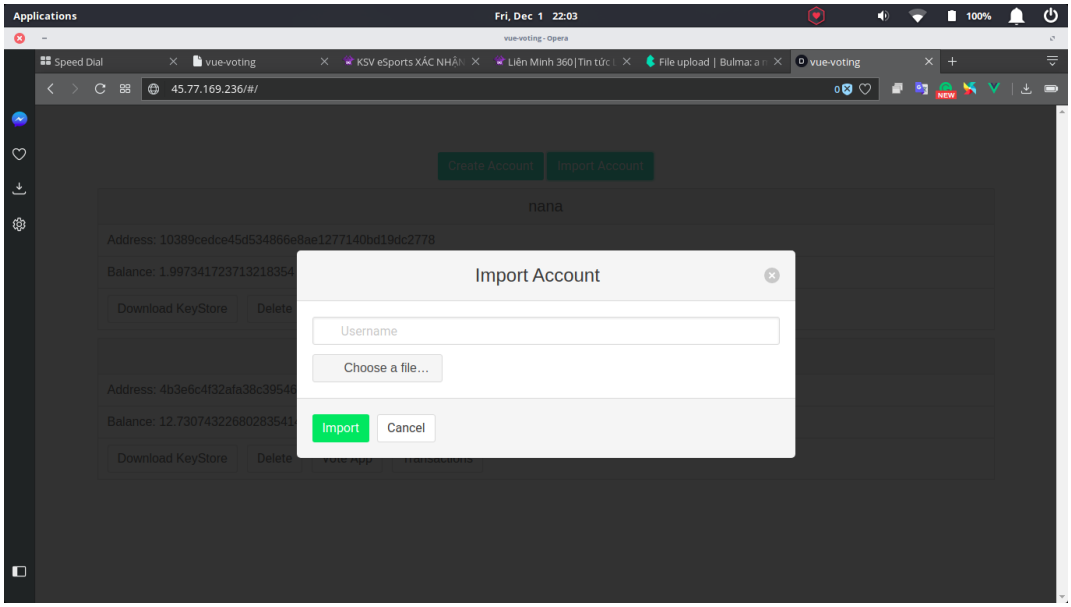
Câu lệnh trên chỉ sử dụng khi ở chế độ phát triển và nếu muốn đưa vào triển khai thì bạn phải chạy câu lệnh npm run build, các tệp html, css, js sẽ được sinh tự động để bạn có thể đưa vào ứng dụng phía backend. Cách trên khá bất tiện nên trong phiên bản tiếp theo sẽ có chỉnh sửa để quá trình được tự động hơn. Sau đây là một vài hình ảnh của ứng dụng:



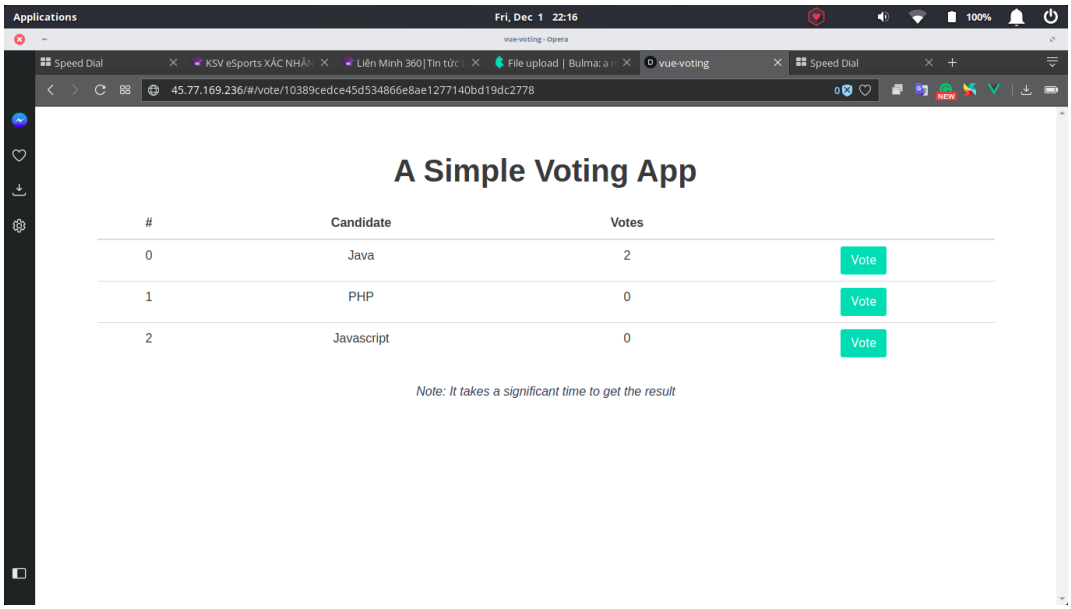
Hình 3: Trang chủ ứng dụng



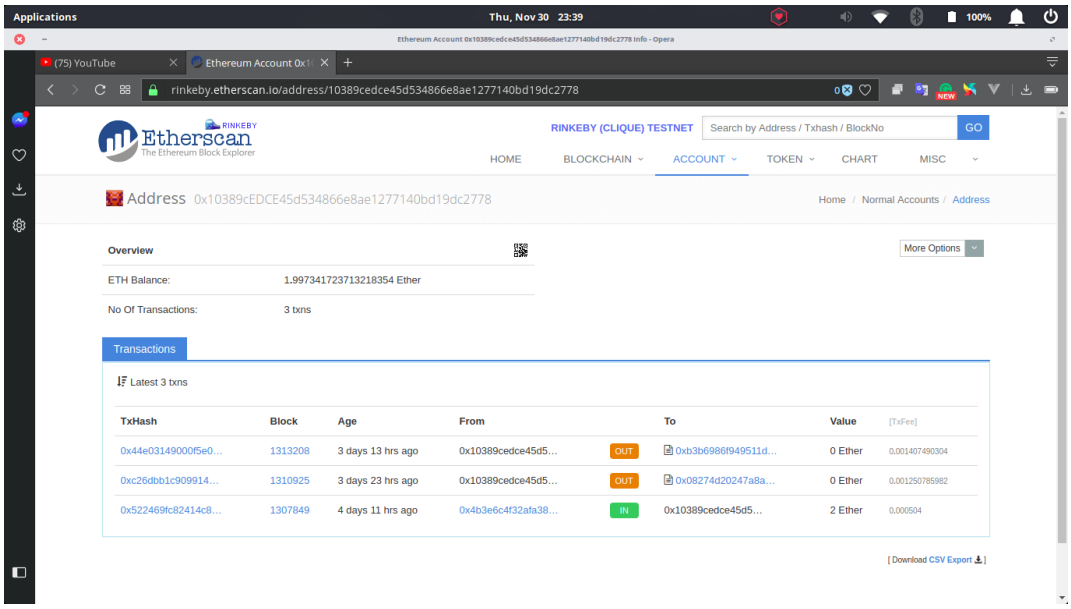
Hình 4: Thêm tài khoản



Hình 5: Nhập tài khoản



Hình 6: Ứng dụng bầu cử



Hình 7: Hình ảnh trang cung cấp tra cứu giao dịch

5.3.4 Một số lưu ý rút ra từ ứng dụng

Phần này được viết thêm và sẽ hữu ích với bạn nếu bạn định sử dụng Ethereum trong mạng thật và chi tiêu cũng như đầu tư vào Ethreum. Nếu bạn đọc mã nguồn của chương trình cũng như có thể suy đoán ra, chương trình trên hoàn toàn có thể có được khóa bí mật của bạn và đó là một rủi ro nếu bạn sử dụng đồng ether thật. Cụ thể với trang `https://www.blockchain.com`, bạn cũng có thể tạo tài khoản một cách trực tuyến mà không phải cài đặt gì, nhận tiền và chuyển tiền

và bạn cũng có thể bị mất nó bất kì lúc nào. Bạn dùng nó và chỉ mong nhà cung cấp dịch vụ không là gì với thông tin cá nhân của mình, cũng như các dịch vụ tập trung khác mà bạn đang sử dụng. Vì vậy nếu muốn an tâm hoàn toàn, bạn nên cài đặt các ví được cung cấp một cách chính thống như https://github.com/ethereum/mist/releases. Bạn vẫn có nguy cơ bị tấn công nhưng ít nhất bạn biết bạn đang giữ tài sản của mình.

6 Tài liệu tham khảo

[1] *www.blockchain.com*

[2] Blockchain, *www.wikipedia.org* (ngày 24 tháng 10 năm 2017)

[3] *www.bitcoin.com*

[4] Ethereum, *www.wikipedia.org* (ngày 11 tháng 10 năm 2017)

[5] *Introducing Ethereum and Solidity*, Chris Dannen

[6] *Ethereum Yellow Paper* - yellowpaper.io, Gavin Wood

[7] *www.rinkeby.io*

[8] Ethereum JavaScript API, *www.web3js.readthedocs.io/en/1.0*

[9] *https://vuejs.org*