**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN HỆ THỐNG THÔNG TIN**

**ĐẶNG ANH KIỆT – PHẠM VĂN HẬU**

**NGHIÊN CỨU VÀ ỨNG DỤNG CÔNG NGHỆ BLOCKCHAIN VÀO GIAO DỊCH**

**BẤT ĐỘNG SẢN**

**TPHCM, 2020**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN HỆ THỐNG THÔNG TIN**

**ĐẶNG ANH KIỆT – 1512269**

**PHẠM VĂN HẬU – 1512154**

**NGHIÊN CỨU VÀ ỨNG DỤNG CÔNG NGHỆ BLOCKCHAIN VÀO GIAO DỊCH**

**BẤT ĐỘNG SẢN**

**KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT**

**GIÁO VIÊN HƯỚNG DẪN**

**Thái Hùng Văn**

DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Ký hiệu | Nguyên nghĩa |
| 1 | PKC | Mật mã hóa công khai |
| 2 | PDF | Định dạng tài liệu di động |
| 3 | GIT | Phần mềm quản lí mã nguồn phân tán |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

DANH MỤC CÁC BẢNG BIỂU VÀ HÌNH ẢNH

[Hình 1. 1 Cấu tạo của một block 3](#_Toc43631088)

[Hình 1. 2: Cấu tạo của một Genesis Block 3](#_Toc43631089)

[Hình 1. 3: Mối liên kết giữa 2 block 4](#_Toc43631090)

[Hình 1. 4: Sự tương quan giữa dấu vân tay và mã Hash 4](#_Toc43631091)

[Hình 1. 5: Giao dịch mua truyền thống 6](#_Toc43631092)

[Hình 1. 6: Một block bị kẻ gian sửa đổi 7](#_Toc43631093)

[Hình 1. 7: Hệ thống mạng lưới P2P 8](#_Toc43631094)

[Hình 1. 8: Hệ thống mạng P2P phát hiện sự tấn công 9](#_Toc43631095)

[Hình 1. 9: Mạng lưới P2P không cấu trúc 10](#_Toc43631096)

[Hình 1. 10: Một block đầy đủ thông số cần thiết 12](#_Toc43631097)

[Hình 1. 11: Một ví dụ về mã băm thỏa và không thỏa việc tạo ra mã băm của hệ thống 13](#_Toc43631098)

[Hình 1. 12: Cây Merkle của các giao dịch A, B, C, D 17](#_Toc43631099)

[Hình 3. 1: github 17](#_Toc32744922)

# MỤC LỤC

Table of Contents

[MỤC LỤC 3](#_Toc45348172)

[PHẦN 1: MỞ ĐẦU 1](#_Toc45348173)

[1.1 Đặt vấn đề 1](#_Toc45348174)

[1.2. Mục đích của nghiên cứu: 1](#_Toc45348175)

[1.3. Phạm vi của nghiên cứu: 1](#_Toc45348176)

[1.4. Giả định được sử dụng: 1](#_Toc45348177)

[2. Mục đích nghiên cứu 2](#_Toc45348178)

[3. Đối tượng và phạm vi nghiên cứu 2](#_Toc45348179)

[4. Nội dung và nhiệm vụ nghiên cứu 2](#_Toc45348180)

[5. Phương pháp nghiên cứu 2](#_Toc45348181)

[6. Đóng góp của đề tài 2](#_Toc45348182)

[7. Kết cấu của đề tài 2](#_Toc45348183)

[PHẦN 2: Kiến thức nền tảng 2](#_Toc45348184)

[2.1 BLOCKCHAIN 2](#_Toc45348185)

[2.1.1 Blockchain là gì? 2](#_Toc45348186)

[2.1.2 Hash: 4](#_Toc45348187)

[2.1.2 SHA-256: 5](#_Toc45348188)

[2.1.3 Sổ cái bất biến(Immutable Ledger): 6](#_Toc45348189)

[2.1.4 Mạng lưới phân tán P2P(Distributed P2P Network): 7](#_Toc45348190)

[2.1.5 Mining: 12](#_Toc45348191)

[2.1.5.1 Cách một Mining làm việc với Nonce: 12](#_Toc45348192)

[2.1.6 Hệ thống chịu lỗi Byzantine: 13](#_Toc45348193)

[2.1.7 Cơ chế đồng thuận (Consensus Protocol): 15](#_Toc45348194)

[2.1.8 Mật mã hóa công khai (Public Key Cryptography): 16](#_Toc45348195)

[2.1.9 Merkle Tree: 16](#_Toc45348196)

[2.2: ETHEREUM 17](#_Toc45348197)

[2.2.1 Khái niệm: 17](#_Toc45348198)

[2.2.2 Tài khoản(Account): 17](#_Toc45348199)

[2.2.3 Giao dịch(Transaction): 18](#_Toc45348200)

[2.2.4 Gas: 18](#_Toc45348201)

[2.2.5 Bộ nhớ(memory), lưu trữ(storage), ngăn xếp(stack): 19](#_Toc45348202)

[2.2.6 Gọi tin nhắn(Message calls): 19](#_Toc45348203)

[2.2.7 Nhật kí (Logs): 20](#_Toc45348204)

[2.3: SOLIDITY 20](#_Toc45348205)

[2.3.1 Solidity 20](#_Toc45348206)

[2.3.2 Contruct Structure 20](#_Toc45348207)

[2.3.3 Types 24](#_Toc45348208)

[PHẦN 3: Thiết Kế Hệ Thống 26](#_Toc45348209)

[3.1 Cấu trúc 26](#_Toc45348210)

[3.1.1 Lựa chọn thiết kế 26](#_Toc45348211)

[3.1.2 Giải pháp 26](#_Toc45348212)

[3.2 Giải pháp lưu trữ 26](#_Toc45348213)

[3.2.1 IPFS(Interplanetary File System) 27](#_Toc45348214)

[3.2.2 Ethereum Swarm 27](#_Toc45348215)

[3.2.3 So sánh IPFS và Ethereum Swarm 27](#_Toc45348216)

[3.2.4 Kết hợp và thực thi 27](#_Toc45348217)

[PHẦN 4: THẢO LUẬN 27](#_Toc45348218)

[4.1 On-chain và Off-chain 27](#_Toc45348219)

[4.2 Cải tiến Hợp đồng thông minh 27](#_Toc45348220)

[4.3 Cải tiến sự kết hợp 27](#_Toc45348221)

[PHẦN 5: KẾT LUẬN 27](#_Toc45348222)

[PHẦN 6: PHỤ LỤC 27](#_Toc45348223)

[TÀI LIỆU THAM KHẢO 27](#_Toc45348224)

# PHẦN 1: MỞ ĐẦU

## 1.1 Đặt vấn đề

Cho đến thời điểm hiện tại thì việc mua bán bất động sản vẫn theo hình thức truyền thống. Đó là trong quá trình mua bán bất động sản vẫn tồn tại một bên thứ ba là người môi giới, giấy tờ pháp lý, chứng từ, bên thứ ba tin cậy, ngân hàng. Bên cạnh đó, quá trình để hoàn tất việc mua bán bất động sản thường tốn khá nhiều thời gian, và một số thông tin không minh bạch và toàn vẹn, vấn đề trốn thuế vẫn còn xảy ra cũng như cung cấp những thông tin sai lệnh cho cơ quan thuế.

Quá trình mua bán diễn ra chậm cơ bản là do nhiều lý do như quá trình xác nhận lặp đi lặp lại, quá trình xác nhận được kí trên giấy và cần được gửi đi xác nhận. Xác nhận tài liệu làm thủ công, có thể xảy ra sai xót, sửa chữa. Hoàn toàn có thể xảy ra sự gian lận và tham nhũng. Tấc cả những quá trình này làm chi phí giao dịch cao và tốn thời gian.

Một ngôi nhà, một miếng đất nếu có giá phải chăng thì rất quan trọng đối với người mua, đặt biệt là những người có thu nhập trung bình và thấp.

## 1.2. **Mục đích của nghiên cứu:**

Đưa ra một bản hợp đồng thông minh dựa vào nền tảng blockchain. Nhằm mục đích giảm chi phí giao dịch, không cần vai trò của bên trung gian tin cậy, và đảm bảo được sự minh bạch trong quá trình giao dịch cũng như chống tham nhũng.

## 1.3. Phạm vi của nghiên cứu:

Mô hình phát triển được dựa trên: Cơ sở dữ liệu MySQL, Truffle framework, Java Servlet, JSP, Web3j, Ethereum test, Solidity.

## 1.4. Giả định được sử dụng:

Chính phủ chấp nhận việc sử dụng tiền điện tử.

## 2. Mục đích nghiên cứu

## - Tìm hiểu về các công nghệ mới để áp dụng vô dự án

## 3. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiêng cứu là những người có nhu cầu mua nhà đất.

## 4. Nội dung và nhiệm vụ nghiên cứu

## 5. Phương pháp nghiên cứu

## 6. Đóng góp của đề tài

## 7. Kết cấu của đề tài

# PHẦN 2: Kiến thức nền tảng

# 2.1 BLOCKCHAIN

## Blockchain là gì?

- Khái niệm blockchain được đề xuất từ hai nhân vật Stuart Haber và W.Scott Stornetta. Bài báo “How to Time-Stamp a Digital Document” (Haber, n.d.) được giới thiệu vào năm 1991 đã đễ xuất sử dụng chứ ký số, băm tài liệu để tăng khả năng bảo mật cho tài liệu truyền đi trên internet. Tất cả những kỹ thuật được nêu trong bài báo trên được sử dụng trong hệ thống blockchain hiện tại.

- Blockchain là một danh sách các block được lưu liên tục theo thời gian, các block này được liên kết với nhau và bảo mật sử dụng mật mã. (Wikipedia, 2019).

- Một block cơ bản bao gồm 3 thành phần chính: dữ liệu, mã hash của block liền trước, mã hash hiện tại. Ngay tại lúc này, ta xem hash như một dấu vân tay, nghĩa là một mã hash là duy nhất và đại diện cho một cá thể.



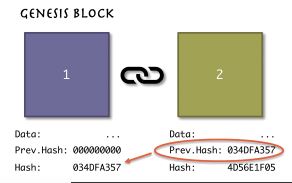
Hình 1. 1 Cấu tạo của một block

- Một blockchain luôn tồn tại một block đầu tiên, còn được gọi là Genesis Block. Block này sẽ không bao giờ thay đổi, và chứa mã hash của hash liền trước là dãy số toàn bit 0.



Hình 1. 2: Cấu tạo của một Genesis Block

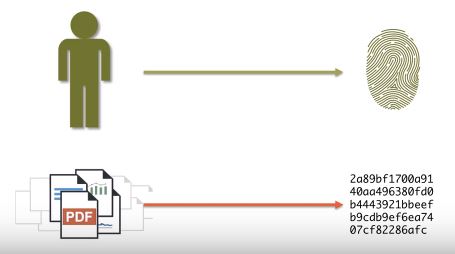
- Các block liền kề nhau trong blockchain có mối quan hệ với nhau qua mã hash. Mã hash liền trước của block hiện tại chính là mã hash của block liền trước nó. Chính sự liên kết này tạo thành một liên kết mắt xích liên tục. Chính vì vậy, nếu có bất kì thay đổi nào của một block bất kì thì sẽ làm thay đổi toàn bộ block trong hệ thống blockchain.



Hình 1. 3: Mối liên kết giữa 2 block

## 2.1.2 Hash:

- Như đã nêu ở nội dung 1.1 thì bên trong mỗi block đều tồn tại một dữ liệu hash, và mã hash đó được ví như là một dấu vân tay riêng biệt. Xác suất của 2 người có cùng dấu vân tay là 1/60.000.000, vì vậy, ta có thể nói dấu vây tay của một người là duy nhất trên toàn thế giới. Sự tương quan của dấu vân tay so với mã hash trong blockchain được mộ phỏng như sau: Con người – Tài liệu, Dấu vân tay – Chuỗi kí tự.



Hình 1. 4: Sự tương quan giữa dấu vân tay và mã Hash

- Chuỗi kí tự phát sinh khi tạo mã hash được sử dụng bởi một thuật toán được gọi là SHA-256.

- Năm yêu cầu của một thuật toán Hash:

1. Một chiều: chỉ có thể đưa vào dữ liệu để lấy mã hash, không thể dùng mã hash để truy xuất ngược lại dữ liệu ban đầu.

2. Sự xác định: luôn cho ra một mã hash duy nhất và giống nhau nếu cùng một dữ liệu ban đầu.

3. Tính toán nhanh: tốc độ phát sinh mã hash phải nhanh cho mọi dữ liệu nhập vào.

4. Hiệu ứng tuyết lỡ: với một thay đổi rất nhỏ trong dữ liệu đầu vào cũng dẫn đến một mã hash thay đổi rất nhiều. Đây là yếu tố quan trọng nhất trong một thuật toán hash.

5. Giải quyết đụng độ: giải quyết được vấn đề hai dữ liệu đầu vào khác nhau nhưng cho ra cùng một mã hash.

## 2.1.2 SHA-256:

- Đây là thuật toán được công bố bởi tổ chức NSA.

- SHA256 là viết tắt của Secure Hash Algorithm 256-bit (Thuật toán băm bảo mật 256 bit) và được sử dụng để bảo mật bằng mật mã.Thuật toán băm mật mã tạo ra các hàm băm không thể đảo ngược và duy nhất.

- Secure Hash Algorithm hay thuật giải băm an toàn là năm thuật toán được chấp nhận bởi FIPS – Tiêu chuẩn Xử lý Thông tin Liên bang, dùng để chuyển một đoạn dữ liệu nhất định thành một đoạn dữ liệu có chiều dài không đổi với xác suất khác biệt cao. Những thuật giải này được gọi là “an toàn” bởi vì, theo nguyên văn của chuẩn mực FIPS 180-2 phát hành ngày 1/8/2002. (brilliant, 2019).

- SHA-256 là một nhánh của hàm băm mật mã SHA-2. Thuật toán SHA-256 tạo ra một mã băm có cố định 256-bit (32-byte) gần như duy nhất. Một chuỗi băm được tạo ra không thể được tính toán trở lại. Điều này làm cho nó phù hợp để xác nhận mật khẩu, thách thức xác thực hàm băm, chống giả mạo, chữ ký số. SHA-256 là một trong những hàm băm kế tiếp đối sau SHA-1 và là một trong những hàm băm mạnh nhất hiện tại.

- Thuật toán này không chỉ dành cho những tài liệu văn bản mà nó còn áp dụng được cho bất kì một tài liệu kỹ thuật số nào, như là âm thanh, video, hình ảnh…

## 2.1.3 Sổ cái bất biến(Immutable Ledger):

- Để mô tả thế nào là một sổ cái bất biến ta xét một ví dụ thực tế về việc mua nhà.

- Việc mua nhà truyền thống thì để mua một ngôi nhà, ta cần có một số tiền đủ với giá trị niêm yết của ngôi nhà, cần ra cơ quan chuyên trách để đăng ký ở hữu ngôi nhà. Bên dưới là hình ảnh mô tả một quá trình mua nhà truyền thống.



Hình 1. 5: Giao dịch mua truyền thống

- Sổ sách chứng nhận chủ quyền nhà đất được ghi chép thủ công vào một cuốn sổ. Nếu quyển sổ đó bị đánh cắp, hoặc bị tẩy xóa thì những thông tin về chủ quyền ngôi nhà sẽ bị mất, điều đó thật nguy hiểm. Để giải quyết vấn đề này ta sẽ sử một một quyển sổ cái điện tử.

- Bất cứ khi nào, một người nào mang giấy tờ đến cơ quan có thẩm quyền để đăng kí quyền sở hữu ngôi nhà của họ, thì khi đó hệ thống sẽ tạo một block lưu tấc cả thông tin đó và thêm vào cuốn sổ cái điện tử đó. Bởi tính chất của blockchain nên rất khó để thay đổi thông tin trong cuốn sổ cái điện tử đó.

- Vấn đề mất cuốn sổ cái sẽ đẽ giải quyết bởi tính chất phi tập trung của blockchain, tính chất đó sẽ được đề cập chi tiết ở phần sau. Bên trên chỉ là một ví dụ nhỏ về ứng dụng sổ cái blockchain trong giao dịch bất động sản. Sổ cái có thể là bất cứ cái gì.

- ***Sổ cái xác nhận quyền sở hữu****:* sổ cái có thể là công ty, một mạng lưới các mối quan hệ, việc làm, sản xuất, một câu lạc bộ.

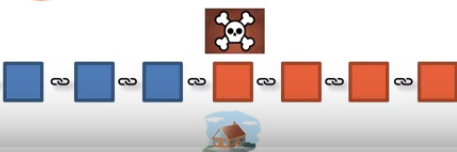
- ***Sổ cái xác nhận danh tính:*** Các doanh nghiệp được ghi trên sổ cái để nhà nước theo dõi sự tồn tài của họ và tình trạng đóng thuế của họ. Các cá nhân cần ghi lại thông tin Sinh tử, Hôn nhân.

- ***Sổ cái xác nhận trạng thái:*** Quyền công dân là một sổ cái ghi lại những người có quyền và có nghĩa vụ đối với một quốc gia. Danh sách bầu cử là một sổ cái. Việc làm là một sổ cái, cho phép người tìm việc tạo hợp đồng để làm việc.

- ***Sổ cái xác nhận thẩm quyền:*** Sổ cái quyết định ai có thể ngồi trong quốc hội một cách hợp lệ. ai có thể truy xuất vào tài khoản ngân hàng, ai có thể làm việc với trẻ em, những người có thể vào các khu vực bị hạn chế. (Chris Berg, n.d.)

## 2.1.4 Mạng lưới phân tán P2P(Distributed P2P Network):

Vấn đề đặt ra là, nếu một block trong sổ cái bị sữa đổi thì hệ thống sẽ bị lỗi. Làm thế nào để khôi phục lại hiện trạng của cuốn sổ cái như lúc chưa bị lỗi. Hệ thống mạng lưới P2P sẽ giải quyết vấn đề trên.

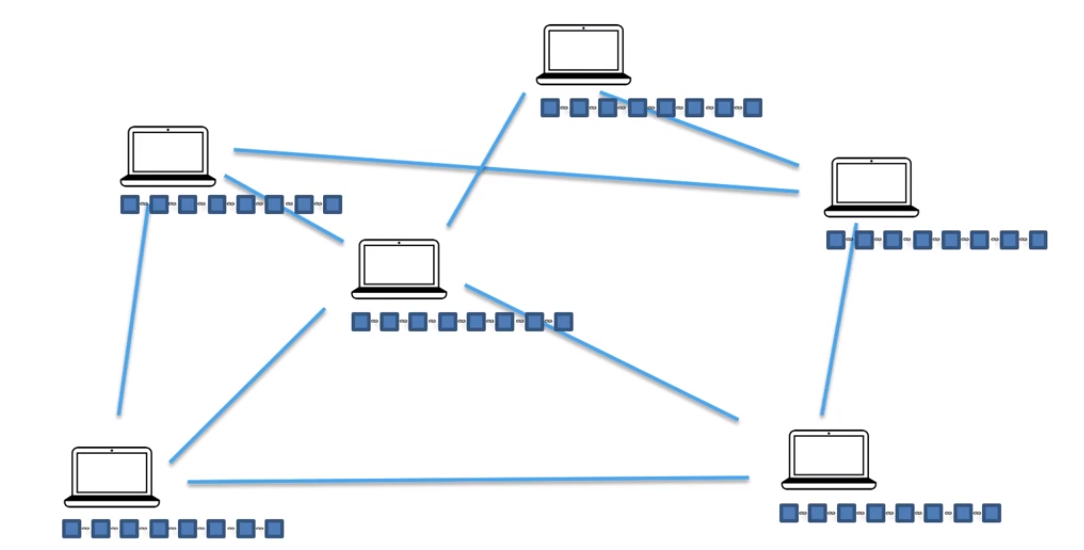


Hình 1. 6: Một block bị kẻ gian sửa đổi

- Hệ thống P2P là hệ thống kết nối giữa các máy tính trực tiếp với nhau, các máy tính trong hệ thống có cùng một cấp độ.

- Một bản sao của cuốn sổ cái sẽ được các máy tính trong hệ thống P2P lưu lại và cập nhật liên tục.

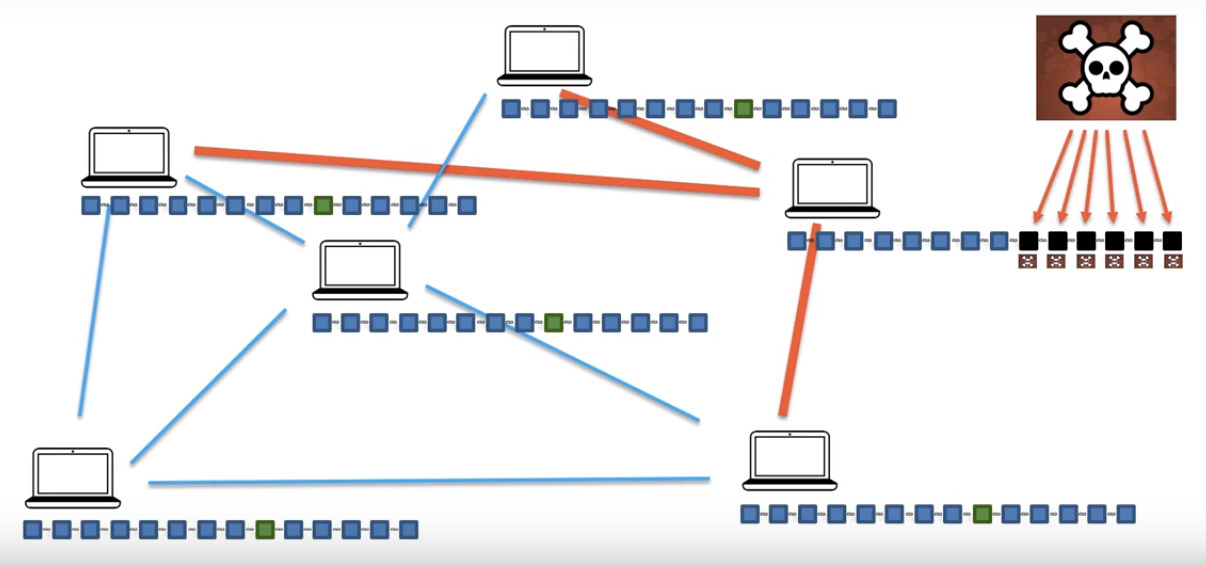
- P2P là một mạng ngang hàng, nó được duy trì bởi một mạng lưới người dùng phân tán ở khắp nới trên thế giới. Mạng này không có người quản trị, không có máy chủ. Mỗi một máy tính trong mạng lưới này vừa đóng vai trò là máy khách, vừa đóng vai trò là máy chủ cho máy khác.



Hình 1. 7: Hệ thống mạng lưới P2P

- Khi một node (máy tính) trong hệ thống bị tấn công, nghĩa là nội dung block trên hệ thống tại máy tính đó đã bị thay đổi, khi đó nó sẽ phát thông báo đến toàn thể các máy tính khác trong hệ thống và ngay lập tức toàn hệ thống sẽ phát hiện sự thay đổi của máy tính bị tấn có nội dung khác với nội dung của các máy còn lại trong hệ thống và khi đó nó sẽ tự động khôi phục lại dữ liệu như cũ.

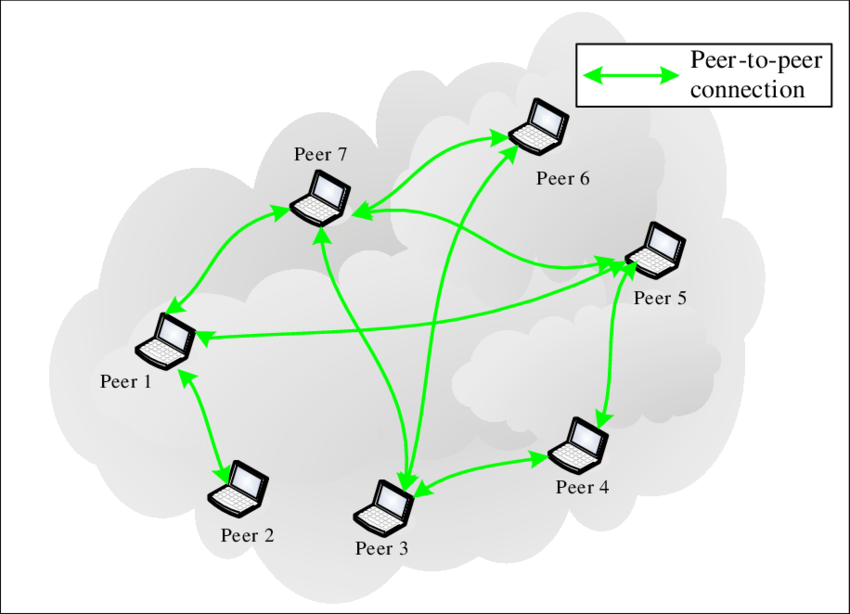
- Để tấn công thành công một máy tính thì kẻ tấn công phải tấn công đồng loạt trên 50% tổng các máy tính trong hệ thống, bởi vì khi đó khi hệ thống so sánh dữ liệu thì sẽ khôi phục theo số đông.



Hình 1. 8: Hệ thống mạng P2P phát hiện sự tấn công

- Theo như tìm hiểu thì kiến trúc của mạng ngang hàng được chia làm 3 loại: không kiến trúc, có kiến trúc và lai.

**- *Mạng ngang hàng(P2P) không có cấu trúc:*** Các nút trên mạng P2P không có cấu trúc không được tổ chức theo bất kì cấu trúc cụ thể nào. Những người tham gia giao tiếp ngẫu nhiên với nhau.



Hình 1. 9: Mạng lưới P2P không cấu trúc

(https://www.researchgate.net/profile/Dan\_Jurca/publication/3199987/figure/fig2/AS:339621113286659@1457983391353/Unstructured-Peer-to-Peer-network-The-receiving-peers-connect-and-retrieve-data-from.png)

- Mặc dù dễ xây dựng hơn, nhưng hệ thống mạng P2P không cấu trúc lại cần sử dụng bộ nhớ và CPU cao hơn, bởi vì một câu truy vấn sẽ được truyền trên cả mạng để tìm ra càng nhiều máy chia sẽ càng tốt. Hệ thống này thể hiện rõ nhược điểm: không có gì đảm bảo tìm kiếm sẽ thành công. Đối với tìm kiếm các dữ liệu phổ biến được chia sẻ trên nhiều máy, tỉ lệ thành công là khá cao, ngược lại, nếu dữ liệu chỉ được chia sẻ trên một vài máy thì xác suất tìm thấy là khá nhỏ. Tính chất này là hiển nhiên vì trong mạng đồng đẳng không cấu trúc, không có bất kì mối tương quan nào giữa một máy và dữ liệu nó quản lý trong mạng, do đó yêu cầu tìm kiếm được chuyển một cách ngẫu nhiên đến một số máy trong mạng. Số lượng máy trong mạng càng lớn thì khả năng tìm thấy thông tin càng nhỏ. Một nhược điểm khác của hệ thống này là do không có định hướng, một yêu cầu tìm kiếm thường được chuyển cho một số lượng lớn máy trong mạng làm tiêu tốn một lượng lớn băng thông của mạng, dẫn đến hiệu quả tìm kiếm chung của mạng thấp. Điều này khiến mạng tràn ngập các truy vấn.

**- *Mạng ngang hàng (P2P) có cấu trúc:*** Các nút trên mạng P2P có cấu trúc một cách có tổ chức, cho phép các nút tìm kiếm các tiệp một cách hiệu quả ngay cả khi chỉ có một số máy trong hệ thống chứa thông tin cần tìm kiếm. Hệ thống này định nghĩa liên kết giữa các nút mạng trong mạng phủ theo một thuật toán cụ thể, đồng thời xác định chặt chẽ mỗi nút mạng sẽ chịu trách nhiệm đối với một phần dữ liệu chia sẻ trong mạng. Với cấu trúc này, khi một máy cần tìm một dữ liệu, nó chỉ cần áp dụng một giao thức chung để xác định nút mạng nào chịu trách nhiệm cho dữ liệu đó và sau đó liên lạc trực tiếp đến nút mạng đó để lấy kết quả.

Trong hầu hết các trường hợp, điều này được giải quyết nhờ dựa vào **hàm băm phân tán** để tra cứu cơ sở dữ liệu.

- Hàm băm phân tán (Distributed Hash Table – DHT): là một lớp (class) của hệ thống phân tán có cấu trúc, cung cấp khả năng tìm kiếm (lookup) tương tự như bảng hash:

+ Là một dạng cấu trúc bảng băm thông thường.

+ Cặp (khóa – key, giá trị - value) được lưu trữ ở DTHs và bất kì node nào cũng có thể truy vấn lấy giá trị một cách hiệu quả thông qua khóa đã cho.

+ Hỗ trợ 3 thao tác: chèn, tìm kiếm, xóa các cặp(key, value).

**- *Mạng ngang hàng (P2P) lai:*** Có một máy chủ trung tâm dùng để lưu trữ thông tin của các máy trạm và trả lời các truy vấn thông tin này. Các máy trạm có vai trò lưu trữ thông tin, tài nguyên được chia sẻ, cung cấp các thông tin về chia sẻ tài nguyên của nó cho máy chủ. Sử dụng các trạm định tuyến để xác định địa chỉ IP của các máy trạm. So với hai kiến trúc còn lại, các mô hình lai thường thể hiện hiệu suất vận hành cao hơn. Chúng kết hợp được các ưu điểm chính của từng phương pháp, mang lại mức độ hiệu quả và phi tập trung đáng kể.

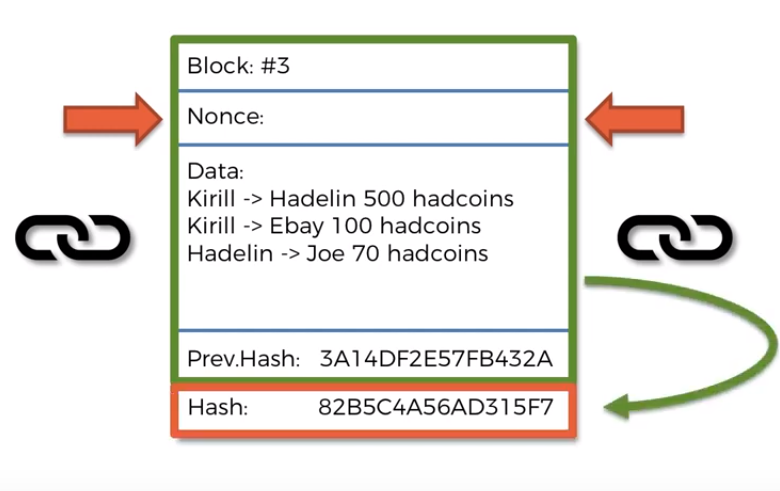
## 2.1.5 Mining:

- Một block trong blockchain lưu trữ không chỉ một giao dịch(transaction) mà lưu trữ rất nhiều giao dịch.

- Như đã đề cập ở mục 1.1 thì một block sẽ lưu trữ 3 thông tin quan trọng là dữ liệu, mã băm hiện tại, mã băm của block liền trước. Một tham số khác bên trong block chưa được nói đến đó là **Nonce**.

- Nonce là từ viết tắt của “Number only used once”.

### 2.1.5.1 Cách một Mining làm việc với Nonce:

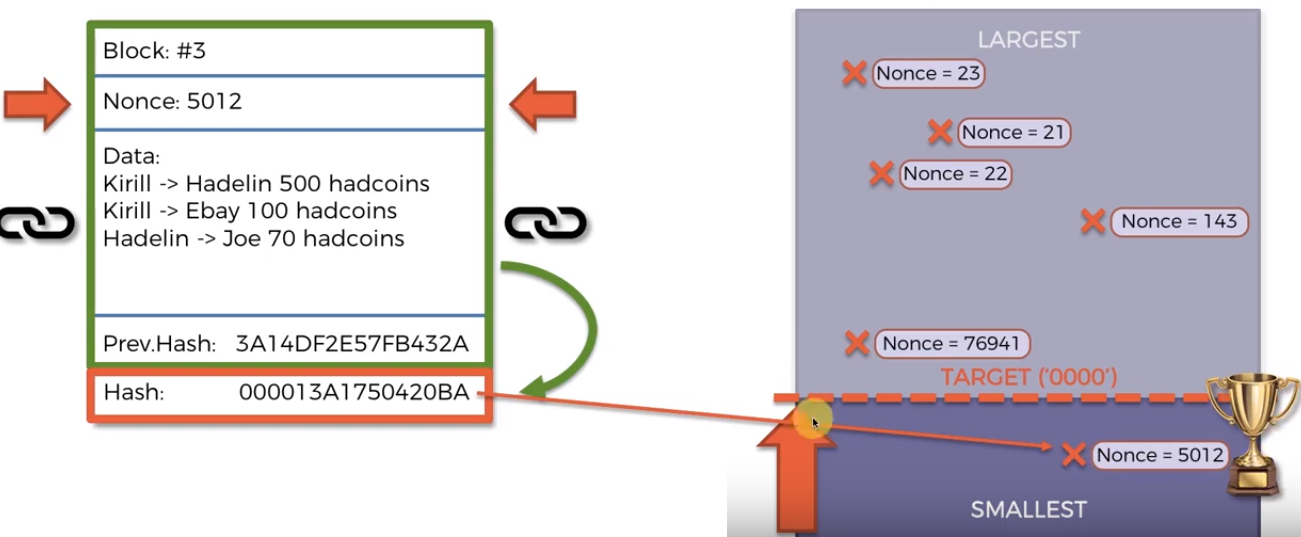


Hình 1. 10: Một block đầy đủ thông số cần thiết

- Hàm hash lúc này nhận vào bốn tham số đó là tiêu đề block, số nonce, dữ liệu giao dịch và mã băm của block trước đó. Trong bốn tham số đó, để đảm bảo sự toàn vẹn dữ liệu và tính bất biến của một hệ thống blockchain thì chỉ duy nhất một tham số nonce là sẽ bị thay đổi liên tục trong lúc tạo ra mã băm mới cho block hiện tại. Và đó là công việc của các thợ đào.

- Số lượng transaction trong hệ thống blockchain rất lớn, vì thế sẽ có rất nhiều block mới được tạo ra để thêm vào cuốn sổ cái blockchain. Để làm giảm số lượng block được thêm vào và làm chậm quá trình thêm block của các thợ đào thì một hệ thống blockchain sẽ có qui định mã băm tạo ra phải đảm bảo theo qui tắc riêng.

- Công việc của thợ đào là thay đổi số nonce liên tục để tạo ra nhiều mã băm sao cho thỏa yêu cầu của hệ thống.



Hình 1. 11: Một ví dụ về mã băm thỏa và không thỏa việc tạo ra mã băm của hệ thống

- Như ví dụ phía trên thì chỉ có mã băm cuối cùng có đủ số lượng bit 0 thỏa với yêu cầu của hệ thống mới được thêm block vào hệ thống.

## 2.1.6 Hệ thống chịu lỗi Byzantine:

- Bài toán các vị tướng Byzantine được đưa ra bởi 3 nhà khoa học máy tính Leslie Lamport, Robert Shostak và Marshall Pease trong một báo cáo khoa học mang tên "The Byzantine Generals Problem" vào năm 1982. Đây là bài toán tổng quát hoá của bài toán 2 vị tướng quân.

- Bài toán các vị tướng Byzantine miêu tả về một nhóm các vị tướng trong đội quân Byzantine (quân đội đế quốc Đông La Mã), tiến hành vây hãm 1 thành phố. Các vị tướng cần trao đổi để đạt được đến 1 thoả thuận về kế hoạch tấn công thành phố đó. Trong trường hợp đơn giản nhất, họ thoả thuận về việc nên tấn công hay rút lui. Một số có thể muốn tấn công, nhưng một số thì lại muốn rút lui, và vấn đề là nếu chỉ có một bộ phận tấn công riêng lẻ, thì họ sẽ gặp thất bại, và đó là kế hoạch tồi tệ hơn việc cùng tấn công hoặc cùng rút lui.

- Blockchain là một cuốn sổ cái được chia sẻ cho mọi thành viên trong một mạng lưới phi tập trung (decentralized). Ở đó, không hề có một bên thứ 3 được tin tưởng để quản lý sự vận hành của nó, mà tự các thành viên của hệ thống phải tương tác với nhau để đi đến một sự đồng thuận (consensus). Hay nói cách khác, một hệ thống Blockchain cần có cách để chịu lỗi Byzantine

- Nếu chúng ta áp dụng bài toán song đề vào trường hợp có sự xuất hiện của blockchain, mỗi tướng sẽ đại diện cho một nút mạng và các nút cần đạt được sự đồng thuận về trạng thái hiện tại của hệ thống. Nói cách khác, phần lớn những người tham gia trong một mạng lưới phân tán phải đồng ý và thực hiện cùng một hành động để tránh một thất bại hoàn toàn.

- Vì vậy, cách duy nhất để đạt được sự đồng thuận trong các loại hệ thống phân tán này là có sự đồng thuận của ít nhất ⅔ hoặc của nhiều hơn các nút mạng trung thực và đáng tin cậy. Điều này có nghĩa là nếu phần lớn nút trong mạng quyết định thực hiện hành động gây hại, hệ thống sẽ dễ bị lỗi và bị tấn công (ví dụ như tấn công 51%).

- Khi Bitcoin, blockchain đầu tiên ra đời, cha đẻ của nó, Nakamoto Satoshi cũng đã giới thiệu một một phương pháp để giải quyết bài toán các vị tướng quân, được gọi dưới cái tên Proof-of-Work (PoW), hay "bằng chứng công việc". (Vaidya, n.d.), (binance, n.d.).

## 2.1.7 Cơ chế đồng thuận (Consensus Protocol):

- Cơ chế đồng thuận của mỗi blockchain có nhiệm vụ đảm bảo cho trạng thái hiện thời của blockchain đó đạt được sự thỏa thuận thống nhất giữa những người tham gia. Cơ chế đồng thuận sẽ quyết định ai là người được phép thêm các block giao dịch mới, và một trong các mục đích cơ bản của nó là để bảo đảm việc chuỗi sẽ không bị ghi đè.

- ***Đồng thuận bằng chứng công việc (Proof of Work):*** là loại đồng thuận mà các thợ đào dùng máy tính của họ để giải bài toán thay đổi hệ số nonce liên tục để tìm ra được mã băm phù hợp để bổ sung block mới vào blockchain hiện tại. Mỗi khi miner đưa ra được lời giải hợp lý, họ có thể đưa ra một block được mạng lưới chấp thuận. Tuy nhiên, hệ thống chỉ chấp nhận chuỗi nào có bằng chứng công việc được tích lũy nhiều nhất (ví dụ như có số lượng block dài nhất, số lượng suy đoán hệ số nonce nhiều nhất, nhiều hash nhất…) làm chuỗi chính thức.

- Nếu một thực thể nào đó kiểm soát được đủ năng lượng tính toán để vượt qua được “chuỗi thật (honest chain)”, khi đó nó có thể viết lại (hoặc sắp xếp lại) blockchain bằng cách khai thác lại trên một block “cũ” thay vì làm việc trên block mới nhất. Hình thức tấn công như vậy, còn gọi là “tấn công 51%”

- ***Đồng thuận Bằng chứng Cổ phần (Proof of Stake):*** Là một phương pháp thay thế trong việc quyết định ai sẽ là người được quyền bổ sung block mới và xác nhận trạng thái hiện thời của blockchain. Thay cho việc các miner sẽ cạnh tranh nhau trong việc tìm lời giải cho các bài toán, với Bằng chứng cổ phần, người tạo ra block tiếp theo sẽ được quyết định dựa theo một số quy trình dựa trên số lượng coin giữ trong ví (cổ phần - stake) của họ.

## 2.1.8 Mật mã hóa công khai (Public Key Cryptography):

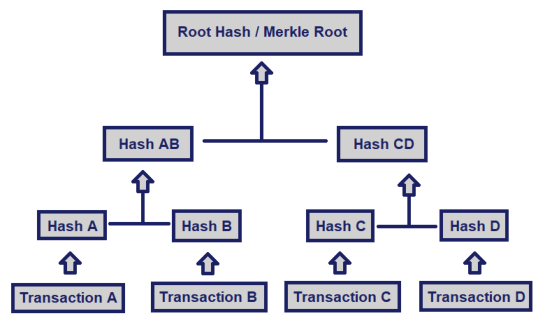
- Mật mã hóa công khai(PKC), là một dạng mật mã hóa cho phép người sử dụng trao đổi các thông tin mật mà không cần phải trao đổi các khóa chung bí mật trước đó. Điều này được thực hiện bằng cách sử dụng một cặp khóa có quan hệ toán học với nhau là khóa công khai và khóa cá nhân.

- Khóa cá nhân phải được giữ bí mật trong khi khóa công khai được phổ biến công khai. Trong 2 khóa, một dùng để mã hóa và khóa còn lại dùng để giải mã. Điều quan trọng đối với hệ thống là không thể tìm ra khóa bí mật nếu chỉ biết khóa công khai.

- Tin nhắn được mã hóa trước khi đưa lên hệ thống blockchain, dữ liệu đều vô nghĩa đối với bất kì ai không có khóa bí mật. Điều đó có nghĩa là tấc cả mọi người trong hệ thống blockchain đều tiếp cận được nội dung của tin nhắn nhưng không thể nào hiểu được nội dung đó.

## 2.1.9 Merkle Tree:

- Cây Merkle là một cấu trúc dữ liệu dạng cây trong đó mọi nút lá được dán nhãn bằng giá trị băm của khối dữ liệu và mọi nút không phải là nút lá được dán nhãn bằng giá trị băm của nhãn của các nút con của nó. Cây băm cho phép xác minh hiệu quả và an toàn nội dung của các cấu trúc dữ liệu lớn.



Hình 1. 12: Cây Merkle của các giao dịch A, B, C, D

# 2.2: ETHEREUM

## 2.2.1 Khái niệm:

Ethereum là một nền tảng phi tập trung dùng để chạy hợp đồng thông minh: những ứng dụng được chạy như một chương trình mà không có bất kì một tác động nào có thể làm nó ngưng hoạt động, không có sự kiểm duyệt, không gian lận, không có sự tác động của bên thứ ba.

Môi trường của smart contract là Ethereum Virtual Machine(EVM). Nó hoàn toàn bị tách biệt. Điều đó có nghĩa rằng là mã của smart contract không được truy cập đến mạng lưới, tập tin của hệ thống, cũng như các tiến trình khác.

## 2.2.2 Tài khoản(Account):

Có 2 loại tài khoản trong Ethereum: tài khoản external và tài khoản contract. Hai loại tài khoản này đều chiếm một không gian lưu trữ như nhau trong Ethereum Virtual Machine. Tài khoản external được quản lí bởi cặp khóa riêng tư và khóa công khai, được sử dụng để người dùng truy cập vào tài khoản. Tài khoản contract được quản lí bởi đoạn mã hợp đồng thông minh. Đoạn mã này được chứa bên trong tài khoản contract. Địa chỉ của tài khoản external được xác định từ khóa công khai, nơi mà tài khoản contract được khởi tạo và triển khai lên mạng lưới Ethereum. Mỗi một tài khoản duy trì một cặp khóa-giá trị. Ngoài ra, mỗi một tài khoản có một số dư tài khoản mà có thể được thay đổi bởi quá trình giao dịch.

## 2.2.3 Giao dịch(Transaction):

Một giao dịch là một tin nhắn được gửi từ một tài khoản này đến tài khoản khác. Nó bao gồm Ether và dữ liệu. Nếu tài khoản đích có chứa mã hợp đồng thông minh, thì đoạn mã đó sẽ thực thi với dữ liệu đầu vào chứa trong tài khoản gửi. Nếu tài khoản đích có địa chỉ là 0(zero) thì một chương trình báo lỗi được khởi tạo. Dữ liệu xuất sau quá trình thực thi được lưu trữ vĩnh viễn trong hệ thống. Dữ liệu của tin nhắn được lưu dưới dạng bytecode trong EVM.

## 2.2.4 Gas:

Gas là một đơn vị đo lường đặt biệt dùng để giới hạn tài nguyên cần thiết cho việc thực thi một đoạn mã một hợp đồng thông minh. Trong khi EVM thực thi đoạn mã thì gas sẽ được tiêu hao dần dần theo một qui tắc cụ thể. Để gửi một giao dịch thì một khoản Ether nhất định phải được trả bằng cách tiêu thụ đơn vị gas. Quá trình khởi tạo giao dịch được sẽ định nghĩa giá gas(gas price)- là đơn vị nhỏ nhất dùng để thanh toán Ether. Các giao dịch mà có gas price cao hơn sẽ dành được nhiều cơ hội thực thi hơn trong một hàng chờ đợi việc thực thi. Khoản thanh toán này nên được trả trước của giao dịch. Nếu như có bất kì một lượng ga nào mất đi hay giao dịch thất bại thì số gas sẽ được hoàn trả lại. Nếu quá trình giao dịch tiêu thụ vượt giới hạn lượng gas thì một chương trình ghi lỗi sẽ khởi tạo và tất cả sự thay đổi trạng thái của dữ liệu liên quan sẽ được khôi phục về trạng thái trước khi thực thi giao dịch. Vì vậy, ta cần có một kế hoạch sử dụng lượng gas khéo léo trong hợp đồng thông minh, bởi vì sau khi xảy ra lỗi sử dụng vượt quá giới hạn gas thì lượng gas sẽ không được trả lại.

## 2.2.5 Bộ nhớ(memory), lưu trữ(storage), ngăn xếp(stack):

Mỗi một tài khoản có một không gian lưu trữ riêng biệt và được duy trì, nơi đó được gọi là **storage**. Storage là một nơi lưu trữ khóa-giá trị. Cả khóa và giá trị được lưu dưới dạng 256-bit kí tự. Không thể liệt kê chúng được vì mỗi khi truy vấn chúng thì hệ thống đều thực hiện hash khóa để tìm ra giá trị tương ứng. Khi truy cập các biến lưu trữ trong storage đều phải tiêu tốn gas. Việc ghi dữ liệu vào khu vực storage thì tốn phí gas nhiều hơn việc đọc. Một tài khoản contract chỉ có quyền truy cập đến duy nhất một vùng chứa dữ liệu trong storage tương ứng của chúng.

Một khu vực vùng nhớ khác trong ethereum là **memory**. Mỗi một tin nhắn gọi tới hợp đồng thông minh đều nhận được khu vực bộ nhớ tạm thời. Memory là bộ nhớ tuyến tính và có thể được xử lý ở cấp độ byte. Mỗi khi memory được mở rộng thì nó sẽ thêm vào một vùng có độ lớn là 256 bit, đồng thời cũng phải tốn một lượng ga tương ứng.

Ethereum Virtual Machine là một bộ máy ngăn xếp(**stack**), chính vì vậy, tất cả các sự tính toán được thực hiện trong một vùng nhớ được gọi là stack. Nó có kích thước lớn nhất là 1024 phần tử và chứa được 256 bits. Truy cập đến stack bị giới hạn đến 16 phần tử đầu tiên. Không thể nào truy cập đến một phần tử bất kì mà không loại bỏ phần tử đầu tiên của stack.

## 2.2.6 Gọi tin nhắn(Message calls):

Một hợp đồng thông minh này có thể gọi một hợp đồng thông minh khác hoặc có thể gửi Ether đến tài khoản external bằng message calls. Message calls giống như một giao dịch: chúng có nguồn tài nguyên, có đích đến, có dữ liệu, Ether, gas, và được trả về là dữ liệu. Một hợp đồng được gọi sẽ được cung cấp một vùng nhớ và có thể truy cập payload – nó được cung cấp một vùng nhớ riêng biệt được gọi là calldata.

Có một loại đặc biệt của message call là – **delegatecall**. Nó cho phép thực thi đoạn code của địa chỉ đích trong ngữ cảnh của lời gọi đến một hợp đồng. Điều này giúp cho lập trình viên có thể chạy nhiều thư viện liên quan trong lúc thực thi.

## 2.2.7 Nhật kí (Logs):

Ethereum cung cấp một cách để lưu lại nhật kí của một giao dịch. Thông tin được lưu lại dưới dạng một cấu trúc đặc biệt và được đánh chỉ mục. Hợp đồng thông minh không thể truy cập nhật kí. Tuy nhiên, nhật kí có thể được truy cập từ bên ngoài blockchain một cách dễ dàng và được mã hóa một cách an toàn. Vì vậy, bất kì một thành viên nào trong hệ thống có thể kiểm tra một nhật kí mà không cần tải toàn bộ blockchain.

# 2.3: SOLIDITY

## 2.3.1 Solidity

- Smart Contract được phát triển bằng ngôn ngữ gọi là solidity, nó là một loại tĩnh, hỗ trợ kế thừa, có thư viện và các loại phức tạp do người dùng định nghĩa. Solidity chịu ảnh hưởng của c++, Python và Javascript và được thiết kế dành riêng cho máy ảo Ethereum.

- Mỗi tệp tin được cấu trúc theo cách sau. Điều bắt buộc là xác định phiên bản pragma giới hạn phiên bản của trình duyệt biên dịch sẽ được sử dụng. Cú pháp của câu lệnh nhập rất giống với JavaScript ES6. Imports nếu có, được theo sau bởi mô tả hợp đồng thông minh. Sau khi biên dịch trình biên dịch tạo Giao diện nhị phân ứng dụng cho hợp đồng. ABI là một bộ siêu dữ liệu mô tả cách tương tác với hợp đồng. ABI thường được lưu trữ trong một biểu mẫu JSON.

## 2.3.2 Contruct Structure

- Hợp đồng trong Solidity tương tự như các lớp trong các ngôn ngữ hướng đối tượng. Mỗi hợp đồng có thể chứa các khai báo các trạng thái biến, hàm, hàm tự ghi, sự kiện, kiểu cấu trúc và kiểu enum. Hợp đồng cũng có thể mở rộng các hợp đồng khác.

**State** **variables**

|  |
| --- |
| prama solidity ^0.4.23  contract Land {  address public real\_estate; // state variable  //  } |

**Functions**

Hàm là các đơn vị thực thi của mã trong một hợp đồng.

|  |
| --- |
| pragma solidity ^0.4.23;  contract Ballot {  function vote(uint proposal) public {  // Function // ...  }  } |

- Có bốn loại khả năng hiển thị cho các hàm và biến trạng thái: bên ngoài, công khai, nội bộ, riêng tư. Mặc định là công khai. Đối với các biến trạng thái bên ngoài là không thể, mặc định là nội bộ. Các hàm bên ngoài là một phần của giao diện hợp đồng. Họ có thể được gọi từ các hợp đồng khác và thông qua các giao dịch, họ không thể được gọi trong nội bộ. Function public có thể được gọi là cả hai bên ngoài và bên trong, theo mặc định getter được tạo cho biến trạng thái công khai. Function intenal và state variable chỉ có thể được sử dụng trong hàmvà những đối tượng kế thừa của nó. Function private và state variable chỉ hiển thị với hàm nơi chúng được xác định.

**Function modifiers**

- Function modifiers rất giống với khái niệm cài đặt. Họ đóng gói hàm và thay đổi hành vi của nó. Hình thành điểm thực hiện hàm xem rất tương tự như macros. Mã hàm đóng gói được dán vào một trình giữ chỗ trong công cụ sửa đổi hàm.

|  |
| --- |
| pragma solidity ^0.4.23;  contract Purchase {  address public seller;  modifier onlySeller() { // Modifier  require(  msg.sender == seller,  "Only seller can call this."  );  \_;  }  function abort() public onlySeller { // Modifier usage  // ...  }  } |

**Events**

- Sự kiện cung cấp giao diện cho tính năng ghi nhật ký EVM. Mỗi sự kiện được ghi lại và có thể được truy cập bên ngoài blockchain sau này.

|  |
| --- |
| pragma solidity ^0.4.23;  contract SimpleAuction {  event HighestBidIncreased(address bidder, uint amount); // Event  function bid() public payable {  // ...  emit HighestBidIncreased(msg.sender, msg.value); // Triggering event  }  } |

**Struct types**

**-** Cấu trúc cho phép xác định các biến loại tùy chỉnh bằng cách nhóm các trường khác nhau.

|  |
| --- |
| pragma solidity ^0.4.23;  contract Ballot {  struct Voter { // Struct  uint weight;  bool voted;  address delegate;  uint vote;  }  } |

**Enum types**

- Việc liệt kê cho phép tạo các loại tùy chỉnh với tập hợp hữu hạn các giá trị được xác định trước.

|  |
| --- |
| pragma solidity ^0.4.23;  contract Purchase {  enum State { Created, Locked, Inactive } // Enum  } |

## 2.3.3 Types

- Solidity là một ngôn ngữ loại tĩnh. Solidity có một hệ thống loại đa dạng, cung cấp các loại cơ bản có thể được kết hợp để tạo ra các loại phức tạp.

**Value types**

- Các biến của các loại giá trị luôn được truyền theo giá trị, tức là được sao chép khi chúng được truyền như các hàm số hoặc trong các biến. Các loại sau đây là loại giá trị:

* Booleans
* Số nguyên - số nguyên có dấu và không dấu có kích thước khác nhau từ int8 đến int256 Số điểm cố định - không được hỗ trợ đầy đủ trong phiên bản 0.4.23, nhưng có thể được khai báo, nhưng không giao
* Địa chỉ - giữ giá trị 20 byte
* Mảng byte có kích thước cố định
* Địa chỉ chữ
* Chữ hợp lý và số nguyên
* Chuỗi ký tự
* Chữ thập lục phân
* Enums
* Các loại chức năng

**Reference types**

- Các loại phức tạp không phải lúc nào cũng vừa với 256 bit cần độ chính xác cao hơn trong các thao tác vì sao chép chúng rất tốn kém. Ngoài ra, các loại tham chiếu có một thuộc tính bổ sung được gọi là vị trí dữ liệu 14. Nó xác định xem biến của loại này được lưu trữ trong bộ nhớ hoặc lưu trữ. Vị trí dữ liệu ảnh hưởng đến hành vi chuyển nhượng (tạo bản sao độc lập hay không). Có một số loại tham chiếu trong Solidity: mảng động, cấu trúc và ánh xạ. Kiểu ánh xạ có thể được coi là một hàm băm được xây dựng theo cách mà mọi khóa có thể tồn tại và trỏ đến giá trị mặc định mà biểu diễn byte là số không. Khóa trong ánh xạ có thể thuộc bất kỳ loại nào ngoại trừ ánh xạ khác, mảng động, hợp đồng, enum hoặc struct, trong khi giá trị có thể thuộc bất kỳ loại nào kể cả các ánh xạ khác.

|  |
| --- |
| pragma solidity ^0.4.23;  contract PropertyEnlistment {  mapping(string => Offer) tenantOfferMap;  } |

Để đọc thêm và các chủ đề nâng cao, tài liệu tham khảo Solidity chính thức2

đã được kiểm tra.

## 2.3.4 Transfer:

- không phải chuyển tiền đến address mà là chuyển tiền đến contract

-

Solidity cung cấp nhiều phương thức để thực hiện việc chuyển ether trong Ethereum giữa các hợp đồng thông minh. Trong phạm vi của khóa luận, chúng ta sẽ tìm hiểu ba phương thức chuyển tiền chính và sự khác nhau giữa chúng.

2.3.4.1 send(amount):

Đối với phương thức send() thì có 2 đặc điểm cần chú ý đó là:

# PHẦN 3: Thiết Kế Hệ Thống

## 3.1 Cấu trúc

### 3.1.1 Lựa chọn thiết kế

### 3.1.2 Giải pháp

#### 3.1.2.1 Đường dẫn

#### 3.1.2.2 Cơ sở dữ liệu

#### 3.1.2.3 Hợp đồng thông minh

#### 3.1.2.4 Tương tác với hợp đồng thông minh từ Java

## 3.2 Giải pháp lưu trữ

Ethereum là một blockchain hoàn hảo để phát triển hợp đồng thông minh. Tuy nhiên, nếu dùng nó để lưu trữ dữ liệu thì khá là tốn kém.

Một đàm phán của việc giao dịch bất động sản được xem là kết thúc khi hai bên mua và bên bán đã kí vào hợp đồng. Thông thường thì việc đó được thực hiện trên giấy. Trên hợp đồng thông minh thì việc kí được số hóa bằng chữ kí điện tử và có thể mô hình hóa thành một tài liệu PDF. Tài liệu pdf này được mã hóa thành một chuỗi hash và giá trị đó là duy nhất trong hệ thống và phải được lưu trữ(1).

Một tài sản bất động sản thì cần nhiều hình ảnh để tăng tính thực tế và tăng trải nghiệm người dùng. Những hình ảnh đó cũng được mã hóa và được lưu trữ(2).

(1) và (2) là hai nhu cầu cơ bản cần được đáp ứng trong hệ thống. Ta đã biết rằng, ethereum thì không hoàn toàn thích hợp cho việc lưu trữ tài liệu với dung lượng lớn. Còn blockchain lại bị giới hạn về kích thước của một block, vì vậy, những tài liệu này không thể lưu trong một block. Chính vì vậy, chúng ta cần tìm kiếm một giải pháp khác cho việc lưu trữ tài liệu trong ethereum để đáp ứng nhu cầu của chúng ta. Bên cạnh đó, giải pháp lưu trữ mới phải đảm bảo được ba yếu tố then chốt của blockchain là không tập trung, không thể giả mạo và có thể phục hồi. Có hai giải pháp cơ bản được tìm thấy đó là IPFS (ipfs, n.d.) và Ethereum Swarm. (ethersphere/swarm, n.d.)

### 3.2.1 IPFS(Interplanetary File System)

Là một hệ thống lưu trữ tài liệu theo phiên bản có thể lưu trữ và theo dõi sự cật nhật phiên bản mới nhất của tài liệu theo thời gian, rất giống với Git. Đồng thời nó cũng định nghĩa cách truyền tài liệu trên mạng biến nó thành một hệ thống phân phối. Giống như giao thức HTTP, IPFS cho phép IPFS enables a new permanent web and augments the way we use existing internet protocols like HTTP.

IPFS (“interplanetary file system”) là một hệ thống phân tán tệp tin ngang hàng. Cùng một thời điểm, nó là giao thức phân phối nội dung. Content address là một đoạn mã băm nằm trên url đại hiện cho nội dung của tệp tin, có nghĩa là nếu trình duyệt của bạn muốn truy cập đến một trang nào đó trên IPFS thì chỉ cần đưa ra mã băm rồi mạng sẽ tìm máy có lưu trữ dữ liệu với cùng mã băm và sau đó tải trang về máy tính đó của bạn. IPFS là một giao thức ngang hàng, không có điểm nào độc quyền. Mỗi điểm trong mạng lưới lưu trữ cục bộ những đối tượng IPFS. Mỗi node connect đến những node khác truyền các đối tượng (các tệp tin và cấu trúc data khác). Mỗi tệp tin được chia vào các khối, và nhận một mật mã vân tay duy nhất được gọi là multihash (multihash là một giao thức cho việc phân biệt những đầu ra từ những hàm băm mã hóa đa dạng được thiết lập tốt). Theo cách như vậy IPFS lặp lại nội dung trùng lặp trên mạng. Kế hợp bảng băm phân tán (là những bảng băm chưa cặp key-value theo một quy tắc giúp dễ tìm kiếm) và Git (biểu đồ chu kì để lưu các thay đổi) IPFS cũng cấp phiên bản nội dung. Mỗi điểm trong IPFS chỉ lưu trữ nội dung mà nó quan tâm. Thêm những chỉ mục trên bảng băm phân tán có thể cung cấp thông tin những gì lưu trữ trên mỗi node. Mỗi node chỉ lưu những tệp tin mà nó mà nó sử dụng, chỉ khi nó kết nối với hệ thống IPFS thì nó mới để lộ ra những thông tin nó lưu trữ. Điều đó có nghĩa là tệp tin không thế sao lưu vào hệ thống IPFS và node có thể bị tắt sau đó, nếu không có node nào yêu cầu dữ liệu nó sẽ lưu lại trên máy cục bộ và sẽ không được sao chép qua mạng. Để đáp ứng nhu cầu này, những người tạo IPFS đã giới thiệu FILECOIN (là một mạng lưới lưu trữ thiết kế để lưu trữ những thông tin quan trọng nhất của nhân loại) một blockchain thay thế để khuyến khích lưu trữ dữ liệu. Kết hợp IPFS và FILECOIN có thể tạo ra một hệ thống tệp tin phân tán đáng tin cậy.

### 3.2.2 Ethereum Swarm

Ethereum Swarm là một nên tản lưu trữ phân tán và dịch vụ phân tán nội dung, một dịch vụ lớp nền tảng của web3 (version 3 of web được để xuất bởi Gavin Wood, đại diện cho tâm nhìn và trọng tâm mới cho các ứng dụng web, chuyển từ các ứng dụng do trung tâm quản lí tới các ứng dụng được xây dựng trên nền tảng phi tập trung). Mục tiêu chính của Swarm là cung cấp một kho lưu trữ công khai Ethereum đủ phi tập trung và dự phòng, đặc biệt là lưu trữ và phân phối mã và dữ liệu ứng dụng phi tập trung, cũng như dữ liệu blockchain. Từ góc nhìn về kinh tế, nó cho phép những người tham gia đóng góp hiệu quả tài nguyên và băng thông để cung cấp các dịch vụ đã nói ở trên cho tất cả những người tham gia. Từ góc nhìn người dùng cuối, Swarm chỉ khác WWW(World Wide Web) việc tải tệp tin lên không đến một một máy chủ cụ thể. Mục đích là cung cấp dịch vụ lưu trữ ngang hàng và cung cấp dịch vụ chống DDOS, không thời gian chêt, không chịu lỗi. Swarm được thiết kế để tích hợp bên trong blockchain Ethereum.

### 3.2.3 So sánh IPFS và Ethereum Swarm

Có nhiều tranh cãi về sự khác nhau của Ethereum Swarm và IPFS, người ta cho rằng Ethereum Swarm đang làm điều vô nghĩa trong khi chúng ta nên hướng đến phát triển IPFS hay FILECOIN. Dưới đây là một số điểm tưởng đồng và khác biệt của hai hệ thống này theo dựa theo bài viết của Viktor Tron, một trong những người sáng lập ra Ethereum Swarm.

- Giống nhau: Cả hai dự án điều nói về cách lưu trữ tệp tin phân tán hiệu quả. Mục tiêu của hai dự án điều cố gắng cung cấp một giải pháp thay thế cho HTTP tập trung và lỗi thời hiện có. IPFS và Ethereum Swarm thực hiện sao chép các tệp tin vào các node, mặc dù IPFS thực hiện kết hợp với Filecoin. Cả hai điều sử sụng content-address (là một đoạn mã được sinh ra khi mã hóa nội dung bên trong một tệp tin bất kì) trong giao thức phân phối dữ liệu và cung cấp tên miền phi tập trung. Mỗi dự án điều đảm bảo rằng:

* Không có thời gian chết
* Độ trễ khi truy xuất thấp

- Khác nhau:

+ IPFS là giải pháp lớn hơn về quy mô, áp dụng, cộng đồng. Trong khi IPFS đã chứng minh bản thân được áp dụng các trường hợp thực tế, Swarm vẫn còn thử nghiệm. Tuy nhiên cả hai dự án được coi trong giai đoạn alpha. IPFS có cơ sở người sử dụng lớn và cộng đồng lớn hơn , nhưng Swarm lại được hưởng lợi từ việc được tích hợp vào hệ sinh thái ethereum và thừa hưởng lợi thế về cơ sở hạ tầng.

+ Từ quan điểm kỹ thuật, có một số sự khác biệt giữa Swarm và IPFS. Hai hệ thống sử dụng mạng ngang hàng khác nhau, IPFS dựa trên libp2p – một biến thể của việc triển khai bittorrent(là giao thức cho phép chia sẻ các tệp tin ngang hàng, trong quá trình truyền tải tệp tin bạn sẽ không lấy dữ liệu trên máy chủ, tệp tin bạn cần sẽ có từ một người khác giống như bạn).

### 3.2.4 Kết hợp và thực thi

3.2.5 vidu

# PHẦN 4: THẢO LUẬN

## 4.1 On-chain và Off-chain

## 4.2 Cải tiến Hợp đồng thông minh

## 4.3 Cải tiến sự kết hợp

# PHẦN 5: KẾT LUẬN

# PHẦN 6: PHỤ LỤC

## TÀI LIỆU THAM KHẢO

binance, n.d. *binance.vision.* [Online]   
Available at: https://www.binance.vision/blockchain/byzantine-fault-tolerance-explained  
[Accessed 18 10 2019].

brilliant, 2019. *https://brilliant.org.* [Online]   
Available at: https://brilliant.org/wiki/secure-hashing-algorithms/

Chris Berg, S. D. &. J. P., n.d. *medium.com.* [Online]   
Available at: https://medium.com/cryptoeconomics-australia/the-blockchain-economy-a-beginners-guide-to-institutional-cryptoeconomics-64bf2f2beec4  
[Accessed 8 10 2019].

ethersphere/swarm, n.d. *ethersphere/swarm.* [Online]   
Available at: https://github.com/ethersphere/swarm  
[Accessed 13 1 2020].

github, n.d. [Online]   
Available at: https://github.com/hauphvn/blockchain-thesis  
[Accessed 30 11 2019].

Haber, S., n.d. *https://www.anf.es/.* [Online]   
Available at: https://www.anf.es/pdf/Haber\_Stornetta.pdf

ipfs, n.d. *ipfs.io.* [Online]   
Available at: https://ipfs.io/  
[Accessed 12 7 2020].

Vaidya, K., n.d. *medium.com.* [Online]   
Available at: https://medium.com/all-things-ledger/the-byzantine-generals-problem-168553f31480  
[Accessed 14 10 2019].

Wikipedia, 2019. *Wikipedia.* [Online]   
Available at: https://en.wikipedia.org/wiki/Blockchain

www.binance.visio, n.d. *binance.visio.* [Online]   
Available at: https://www.binance.vision/blockchain/peer-to-peer-networks-explained  
[Accessed 22 10 2019].

What size the block is?

Chua dich được: IPFS enables a new permanent web and augments the way we use existing internet protocols like HTTP.

Kiến thức bổ sung:

Mỗi node có tấc cả lịch sử của smart contract, lịch sử của all transactions, trạng thái hiện tại của tấc cả smart contract.

- Bạn dùng gas để thanh toán cho bất kì một sự tính toán nào mà bạn cần chạy

- Trong ethereum có ETHER tại sao chúng ta không dùng chúng làm lại dùng gas để pay cho code. Đó là tại vì giá eth ko cố định.

- wei là đơn vị nhỏ nhất của ether

- 10^9 wei = 1 gwei

- 1 gwei = 10^-9 ether

- Gwei còn được gọi là nanoether

- Gas limit con số tối đa mà bạn sẵn sang trả cho một giao dịch. Con so này càng cao thì bạn phải làm nhiều việc hơn để thực thi transaction bằng ether hoặc smart contract.

- gasCost(Ether) = gasPrice \* gasCost(Gas);

- gasCost(Ether) = gasPrice \* min(gasCost, gasLimit);

-transaction{

…. ..

… ..

gasPrice,

gasLimit

}

- Cach lay gasPrice:

Cach 1: <https://ethgasstation.info/api/ethgasAPI.json>

Cach 2: web3.eth.gasPrice();

Cach 3: estimateGas() method of web3