



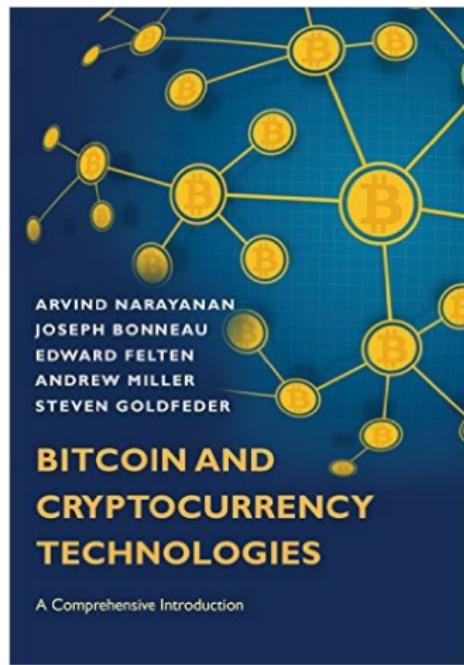
ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

## Nhập môn An toàn thông tin

Giới thiệu sơ lược về Mật mã và Tiền mật mã

Ngày 3 tháng 6 năm 2021

# Tài liệu tham khảo



# Nội dung

① Hàm băm và Chữ ký số

② Tiền mật mã

# Hàm băm mật mã

Hàm băm là một hàm thỏa mãn ba tính chất:

- Đầu vào là một xâu độ dài **bất kỳ**.
- Đầu ra là một xâu độ dài **cố định**. Ta sẽ dùng 256 bit.
- Đây là một hàm tính được một cách **hiệu quả**.

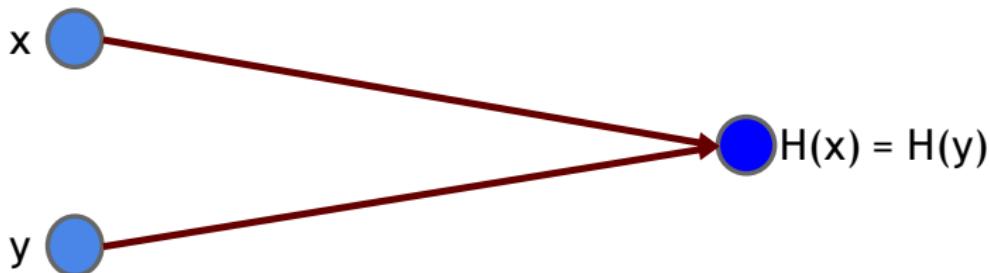
Tính an toàn

- collision-free (kháng xung đột)
- hiding
- puzzle-friendly

## Tính chất 1: Kháng xung đột

Vô cùng khó để tìm hai xâu  $x$  và  $y$  sao cho

$$x \neq y \text{ và } H(x) = H(y).$$



# Ứng dụng của hàm băm kháng xung đột

## 15.04

(Vivid Vervet): April 2015 (Supported until January 2016)

### *md5 Hash*

### *Version*

53c869eba8686007239a650d903847fd ubuntu-15.04-desktop-amd64.iso

6ea04093b767ad6778aa245d53625612 ubuntu-15.04-desktop-i386.iso

487f4a81f22f8597503db3d51a1b502e ubuntu-15.04-server-amd64.iso

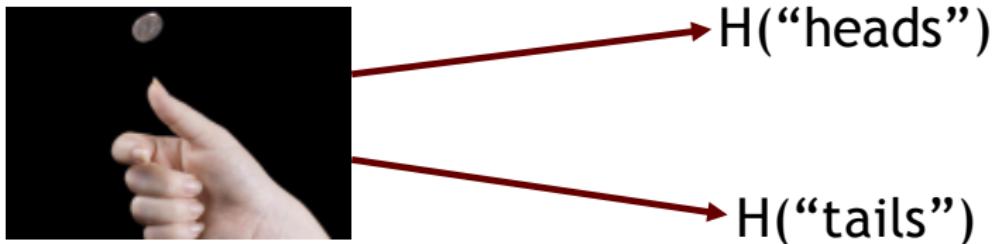
49de7a0ed202d11456126589e2d1db22 ubuntu-15.04-server-i386.iso

fcfba8de8848944705cd200ff76c53cf ubuntu-15.04-snappy-amd64-generic.img.xz

ef2a4951a2e889908a55c980d2bea475 ubuntu-15.04-snappy-armhf-bbb.img.xz

## Tính chất 2: Hiding

- Ta mong muốn tính chất:  
*Cho  $H(x)$ , không thể tìm được  $x$ .*
- Điều này không khả thi khi không gian input nhỏ!



## Tính chất 2: Hiding

Một hàm băm gọi là **hiding** nếu

*khi giữ bí mật giá trị  $r$  được chọn ngẫu nhiên, thì cho  $H(r | x)$  ta không thể tìm được  $x$ .*

**Ứng dụng:** Sơ đồ ủy thác

- Mô phỏng: "Giấu một giá trị trong phong bì dán kín" và "Mở phong bì" sau này.
- Ủy thác một giá trị và tiết lộ nó sau này.

## Sơ đồ ủy thác

API:

- $(com, key) := \text{commit}(\text{msg})$
- $\text{match} := \text{verify}(\text{com}, \text{key}, \text{msg})$

Giấu msg trong phong bì:

- $(com, key) := \text{commit}(\text{msg})$
- sau đó công khai com

Mở phong bì:

- Công khai key, msg
- Ai cũng có thể dùng hàm verify() để kiểm tra tính hợp lệ.

# Tính an toàn của Sơ đồ ủy thác

API:

- $(com, key) := commit(msg)$
- $match := verify(com, key, msg)$

Tính an toàn:

- Hiding: Chỉ cho  $com$ , ta không thể tìm được  $msg$ .
- Binding: Ta không thể tìm được  $msg \neq msg'$  sao cho  $verify(commit(msg), msg') == true$

# Sơ đồ ủy thác

Cài đặt:

- $\text{commit}(\text{msg}) := (\text{H}(\text{key} \mid \text{msg}), \text{H}(\text{key}))$   
với key là một giá trị ngẫu nhiên 256 bit.
- $\text{verify}(\text{com}, \text{key}, \text{msg}) := (\text{H}(\text{key} \mid \text{msg}) == \text{com})$

Tính an toàn dựa trên:

- Hiding: Chỉ cho  $\text{H}(\text{key} \mid \text{msg})$ , không thể tìm được  $\text{msg}$ .
- Binding: Không thể tìm được  $\text{msg} \neq \text{msg}'$  such that  $\text{H}(\text{key} \mid \text{msg}) == \text{H}(\text{key} \mid \text{msg}')$

## Tính chất 3: Puzzle-friendly

Puzzle-friendly:

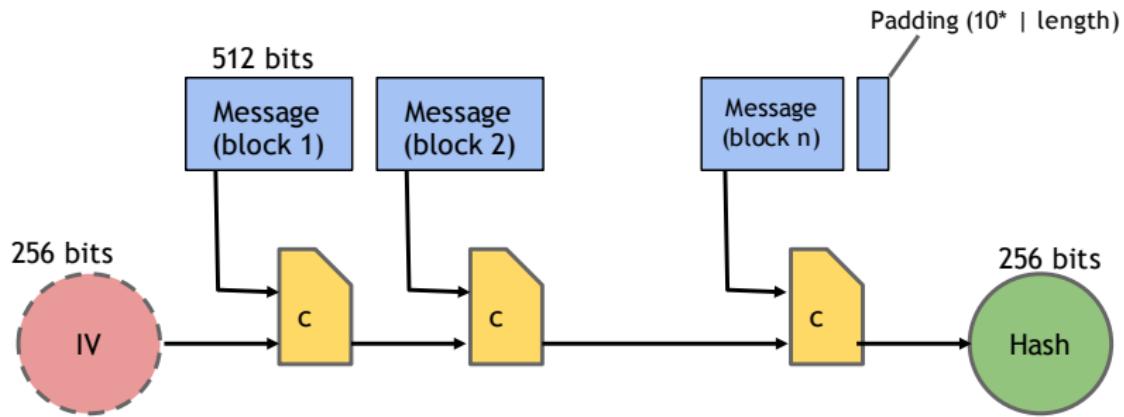
- Với mọi giá trị  $y$  cho trước và  $k$  được chọn ngẫu nhiên, thì vô cùng khó để tìm được  $x$  sao cho  $H(k \mid x) = y$ .

Ứng dụng: Search puzzle

- Cho một “puzzle ID”  $id$ , và một tập giá trị  $Y$ .
- Hãy tìm một “nghiệm”  $x$  thỏa mãn  $H(id \mid x) \in Y$ .

Tính chất Puzzle-friendly chỉ ra rằng không có chiến lược nào tốt hơn việc thử một cách ngẫu nhiên giá trị cho  $x$ .

# Hàm băm SHA-256



## Thảo luận

Thiết kế giao thức để chơi trò chơi dân gian **Oắn tù tì** qua điện thoại.



## Con trỏ băm

Con trỏ băm là

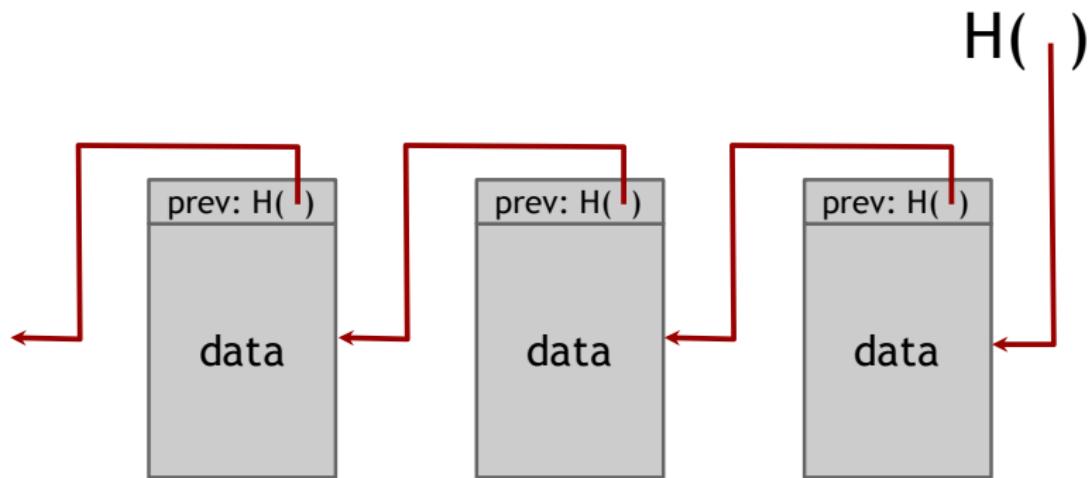
- một con trỏ tới dữ liệu, và
- mã băm của dữ liệu.



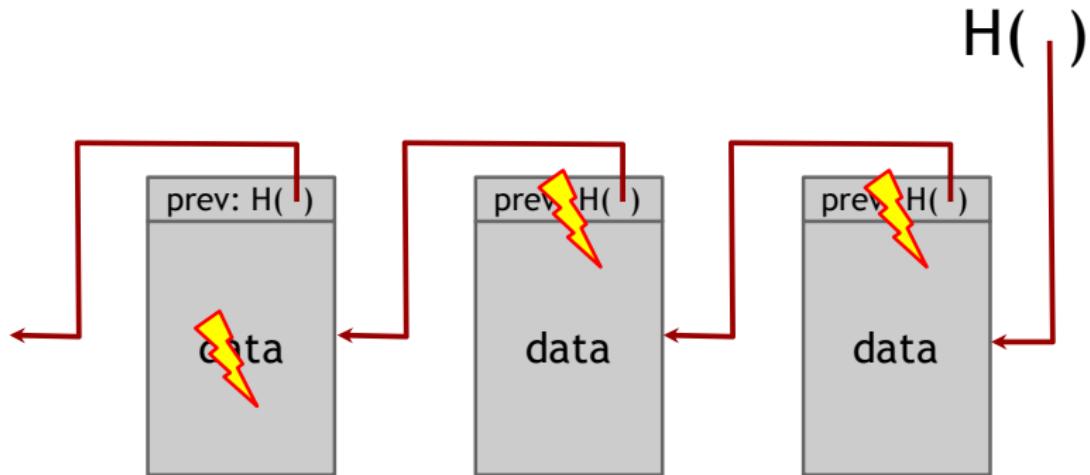
Nếu ta có một con trỏ băm, ta có thể

- lấy dữ liệu, và
- kiểm tra xem liệu dữ liệu có bị sửa đổi.

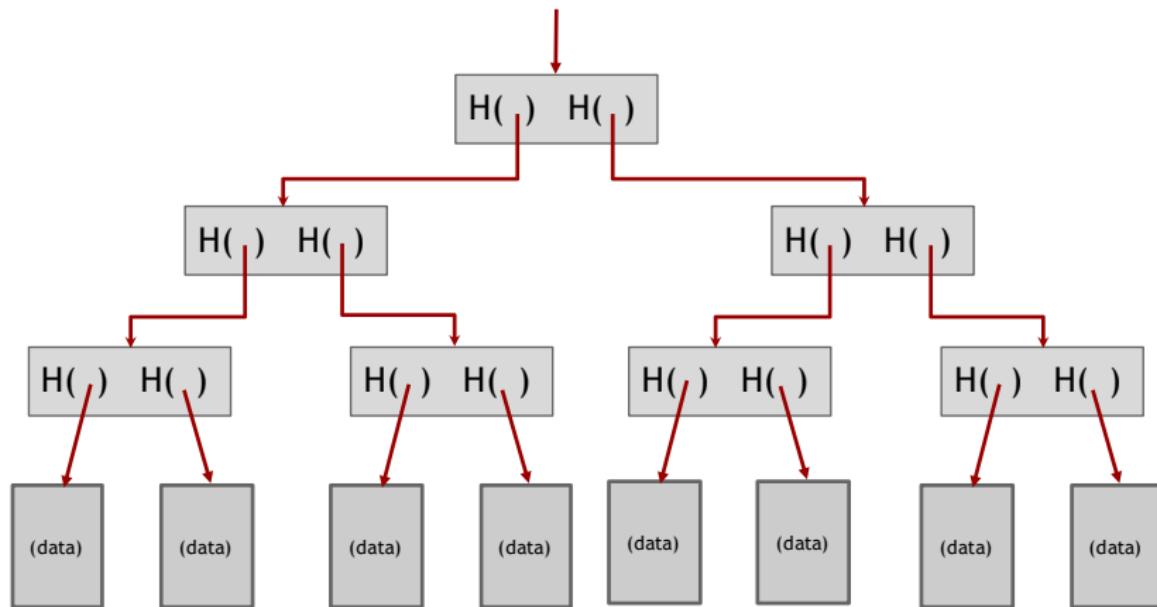
# Danh sách liên kết với con trỏ băm = "blockchain"



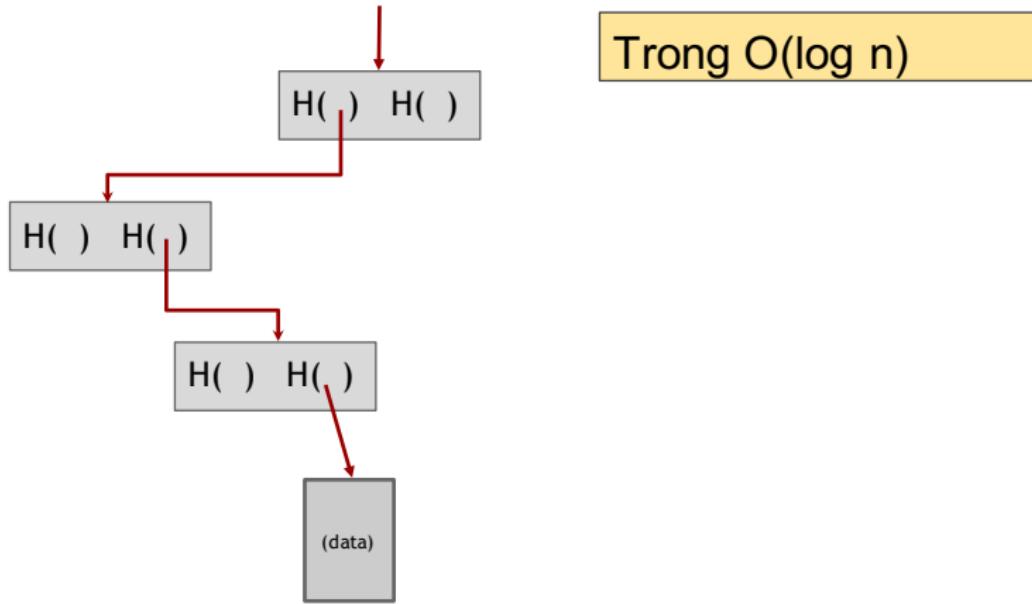
# Phát hiện sửa đổi



# Cây nhị phân với con trỏ băm = “Merkle tree”



Là thành viên của cây?



# Chữ ký số

Ý tưởng:

- Chỉ bạn có thể ký, nhưng ai cũng có thể kiểm tra
- Chữ ký được gắn với một tài liệu cụ thể:  
không thể copy và paste cho tài liệu khác.

API:

- `(sk, pk) := generateKeys(keysize)`
- `sig := sign(sk, message)`
- `isValid := verify(pk, message, sig)`

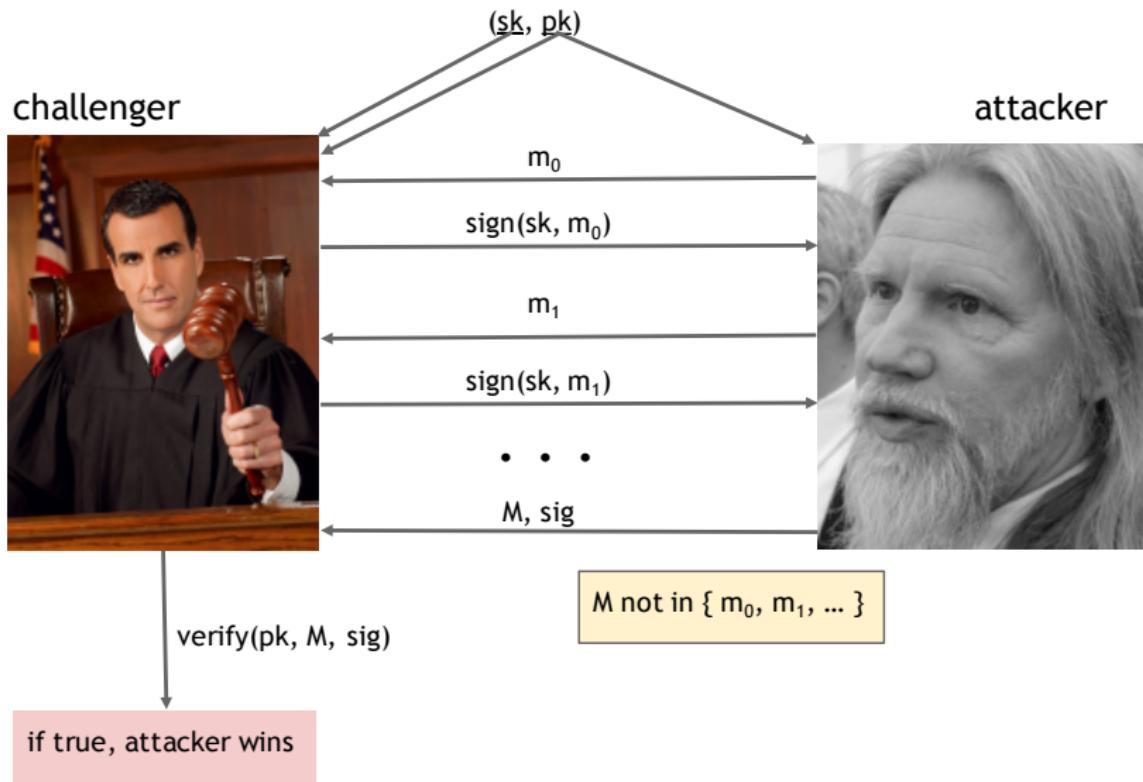
## Tính an toàn của chữ ký số

Kiểm tra đúng chữ ký hợp lệ

```
verify(pk, message, sign(sk, message)) == true
```

Không thể giả mạo chữ ký:

- Dù kẻ tấn công Evil biết pk và có được chữ ký hợp lệ của một số tài liệu (Evil muốn)
- Evil cũng không thể tạo ra chữ ký hợp lệ cho tài liệu mới.



## Thực tế

- Bitcoin sử dụng sơ đồ chữ ký số ECDSA (Elliptic Curve Digital Signature Algorithm)
- Cụ thể, Bitcoin sử dụng đường cong chuẩn secp256k1: có mức an toàn 128 bit
- Kích thước khóa và chữ ký:

Khóa bí mật: 256 bit

Khóa công khai, chưa nén: 512 bit

Khóa công khai, đã nén: 257 bit

Thông điệp được ký: 256 bit

Chữ ký: 512 bit

## Dùng khóa công khai như định danh

- Nếu có sig thỏa mãn

$$\text{verify}(\text{pk}, \text{msg}, \text{sig}) == \text{true}$$

thì ta có thể xem như

pk "thông báo" msg

- Để có thể thông báo trong vai trò pk, ta phải biết khóa bí mật sk.

# Làm thế nào để tạo ra một định danh mới?

Sinh ra một cặp khóa ngẫu nhiên sk, pk:

$$(\text{sk}, \text{pk}) := \text{generateKeys}(\text{keysize})$$

- pk là tên "công khai" mà bạn có thể dùng. Hoặc tốt hơn nếu dùng hash(pk) làm tên công khai.
- sk cho phép tạo ra thông báo.

Bạn hoàn toàn kiểm soát định danh của bạn vì chỉ có bạn biết sk.

# Hệ thống quản lý định danh phi tập trung

- Ai cũng có thể tạo ra định danh mới: bất cứ lúc nào và bao nhiêu cũng được!
- Không cần có hệ thống quản lý trung tâm.
- Trong Bitcoin, định danh được gọi là "địa chỉ".

## Tính riêng tư

- Các địa chỉ không trực tiếp liên kết với định danh trong thế giới thực.
- Nhưng nếu quan sát theo thời gian ta có thể liên kết các hoạt động từ các địa chỉ, và từ đó suy ra mối liên hệ giữa các địa chỉ với đối tượng trong thế giới thực.

# Nội dung

① Hàm băm và Chữ ký số

② Tiền mật mã

GoofyCoin



Goofy có thể tạo ra coin mới

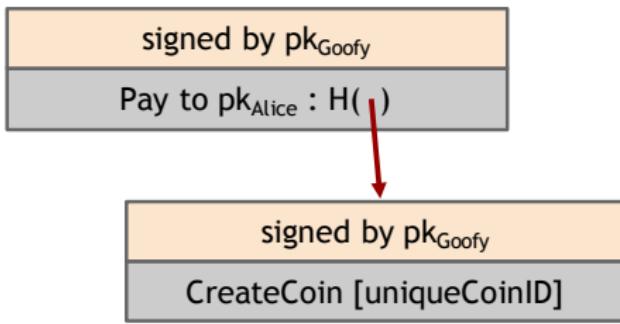
signed by  $pk_{Goofy}$

CreateCoin [uniqueCoinID]

Coin mới thuộc về tôi



# Goofy chuyển coin cho Alice

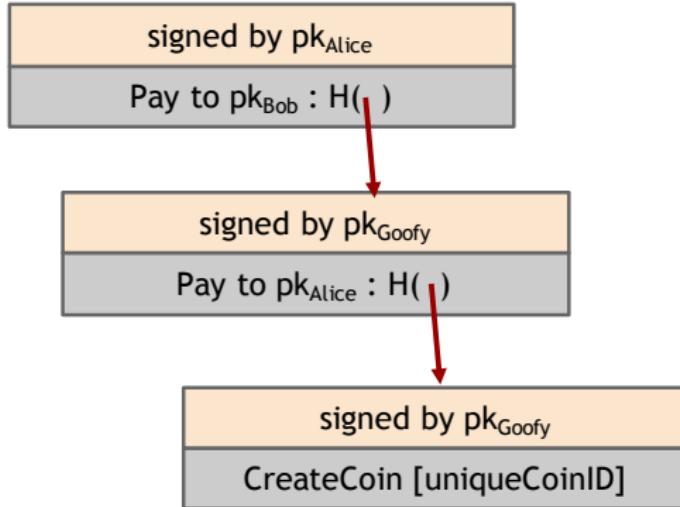


Bây giờ Alice là chủ sở hữu



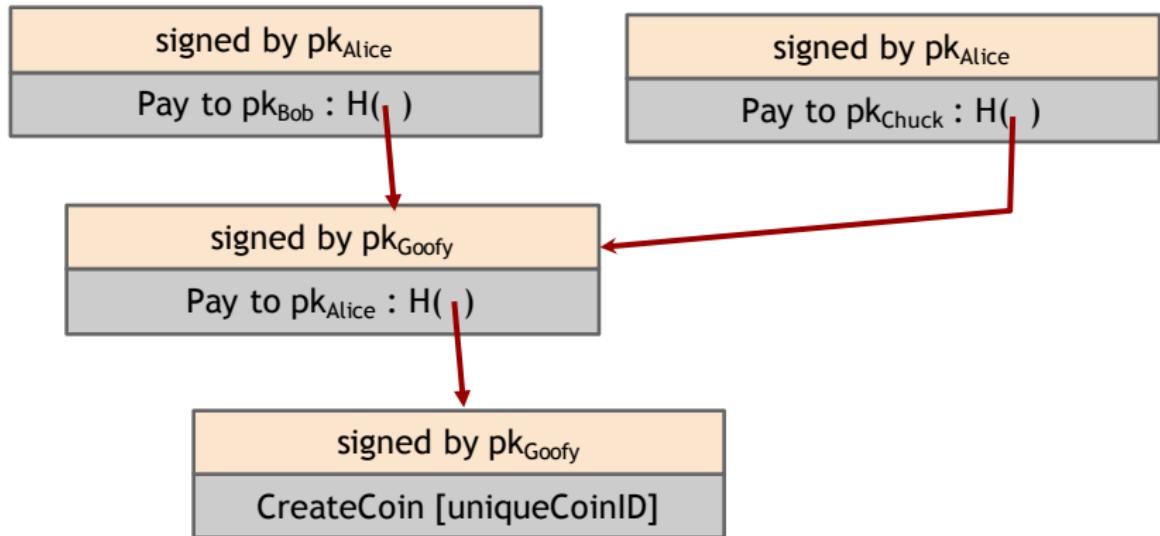
**Hình:** Alice có thể chứng minh mình sở hữu coin bằng cách đưa ra dữ liệu.

## Và Alice chuyển coin cho Bob



**Hình:** Bob có thể chứng minh mình sở hữu coin bằng cách đưa ra dữ liệu.

## Vấn đề double-spending



Hình: Alice tiêu một coin hai lần.

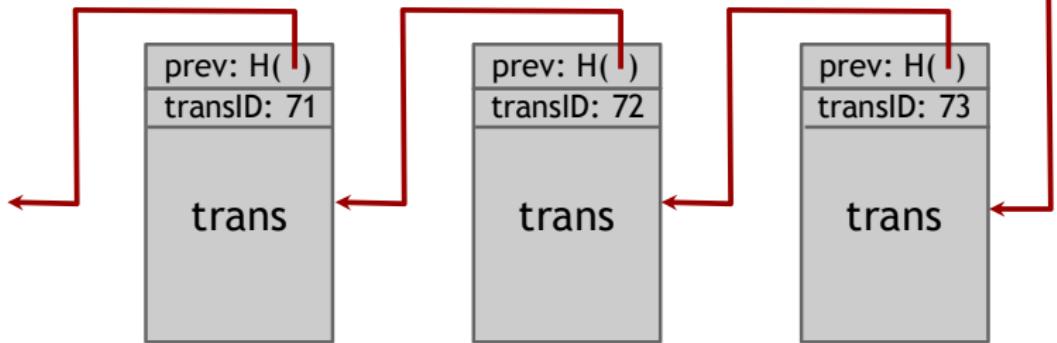
# ScroogeCoin



Scrooge công khai lịch sử của mọi giao dịch  
(một block chain, được ký bởi Scrooge)



$H( )$



## Giao dịch CreateCoins tạo ra coin mới

transID: 73	type:CreateCoins	
coins created		
num	value	recipient
0	3.2	0x...
1	1.4	0x...
2	7.1	0x...

Đúng, vì tôi tạo ra.



coinID 73(0)

coinID 73(1)

coinID 73(2)

# Giao dịch PayCoin

## Tiêu và phá hủy Coin

transID: 73	type:PayCoins			
consumed coinIDs: 68(1), 42(0), 72(3)				
coins created				
num	value	recipient		
0	3.2	0x...		
1	1.4	0x...		
2	-1	0		
signatures				

Hợp lệ nếu:

- coin được tiêu là hợp lệ,
- vẫn chưa được tiêu,
- tổng giá trị out = tổng giá trị in, và
- được ký bởi người sở hữu các coin được tiêu

## Vấn đề của ScroogeCoin

- ScroogeCoin đảm bảo chống được double-spending vì mọi người đều có thể nhìn vào blockchain và đảm bảo các giao dịch là hợp lệ.
- Scrooge không thể tạo ra giao dịch giả vì anh ta không thể giả mạo chữ ký của người khác.
- Nhưng Scrooge có quá nhiều ảnh hưởng:
  - Có thể từ chối công khai một giao dịch nào đó,
  - tạo nhiều coin, hoặc
  - dừng cập nhật blockchain.

## Thảo luận

- Hệ thống ScroogeCoin là hệ thống tập trung. Nó dựa hoàn toàn vào một bên trung gian tin cậy (Scrooge)!
- Xây dựng tiền mật mã hoạt động giống ScroogeCoin nhưng không cần bên trung gian tin cậy.



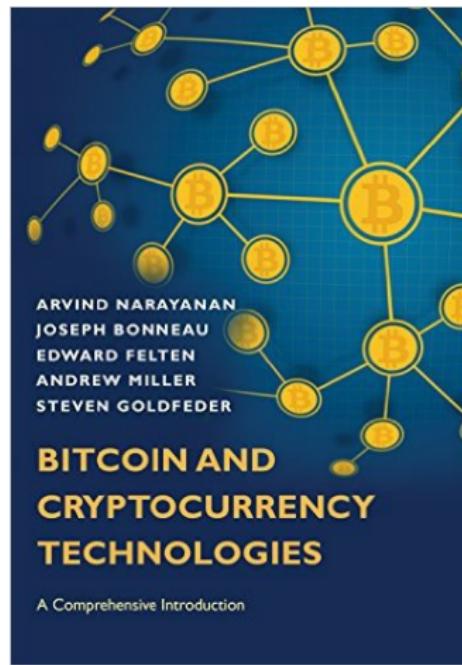
ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# Nhập môn an toàn thông tin

## Giới thiệu sơ lược về Bitcoin

Ngày 3 tháng 6 năm 2021

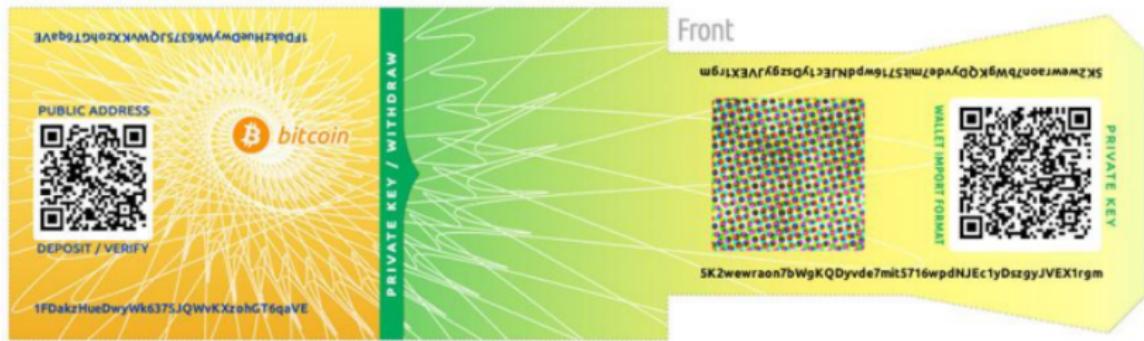
# Tài liệu tham khảo



# Nội dung

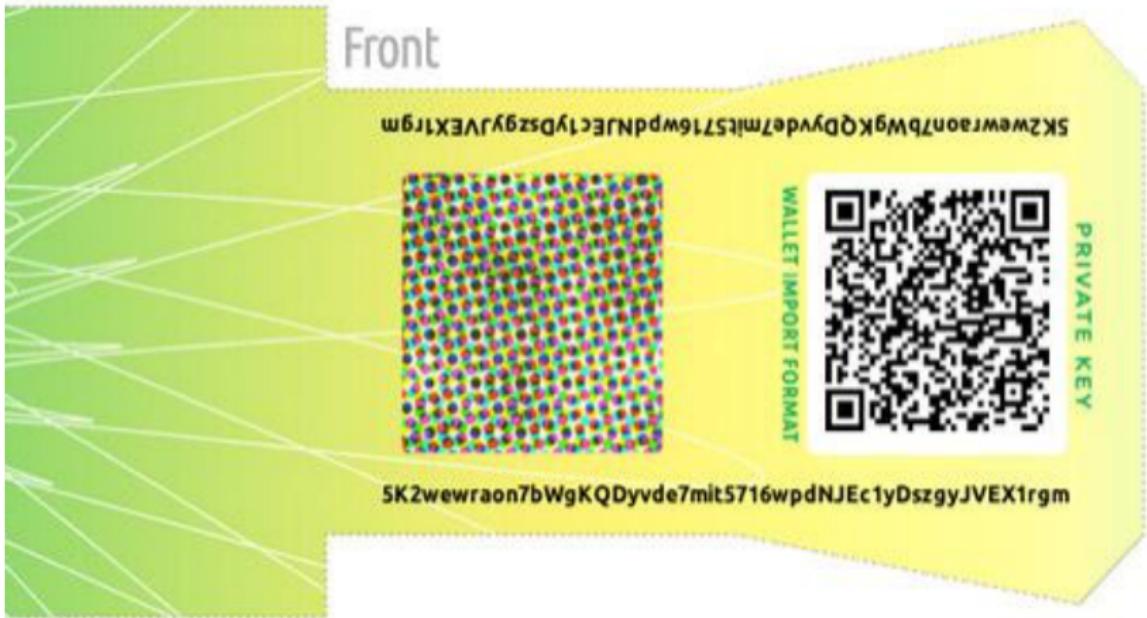
- ① Giao dịch Bitcoin
- ② Cơ chế đồng thuận của Nakamoto
- ③ Cơ chế thưởng và Bằng chứng công việc
- ④ Đào Bitcoin

# Bitcoin Paper Wallet

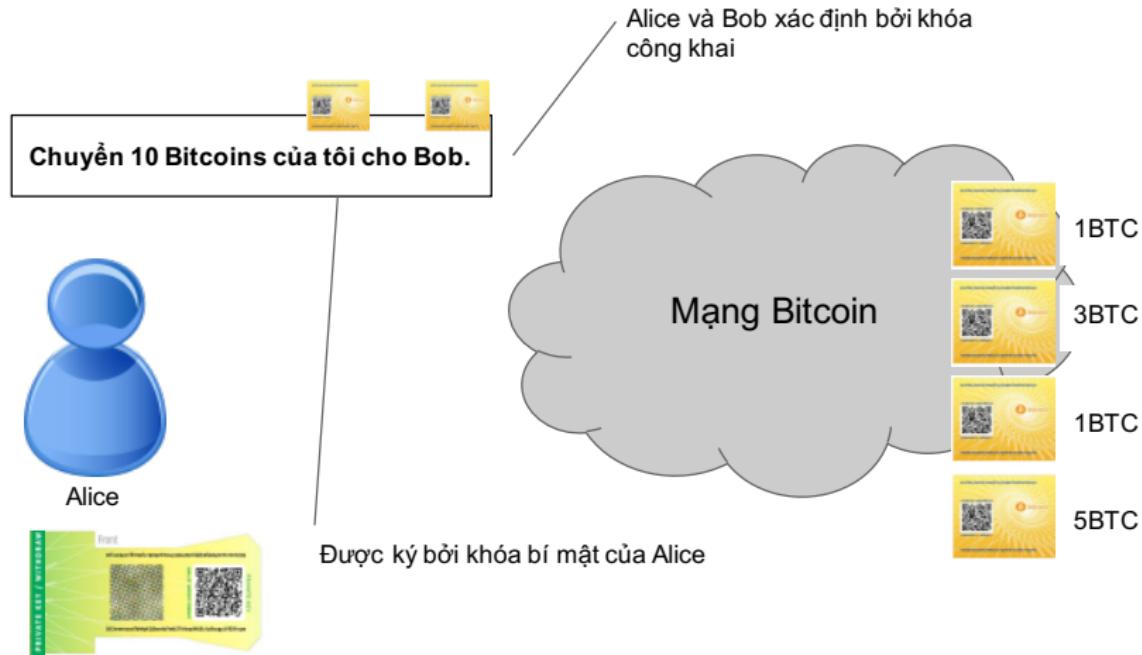


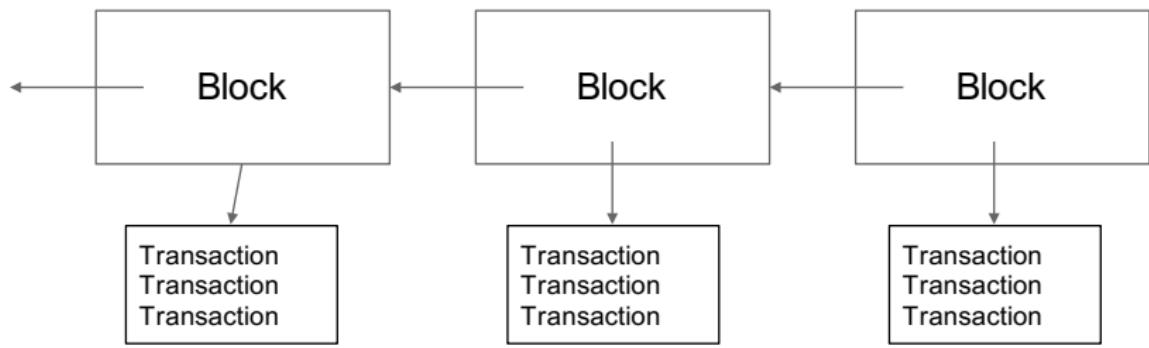
Hình: Tạo ra với <https://bitcoinpaperwallet.com/>

# Khóa bí mật



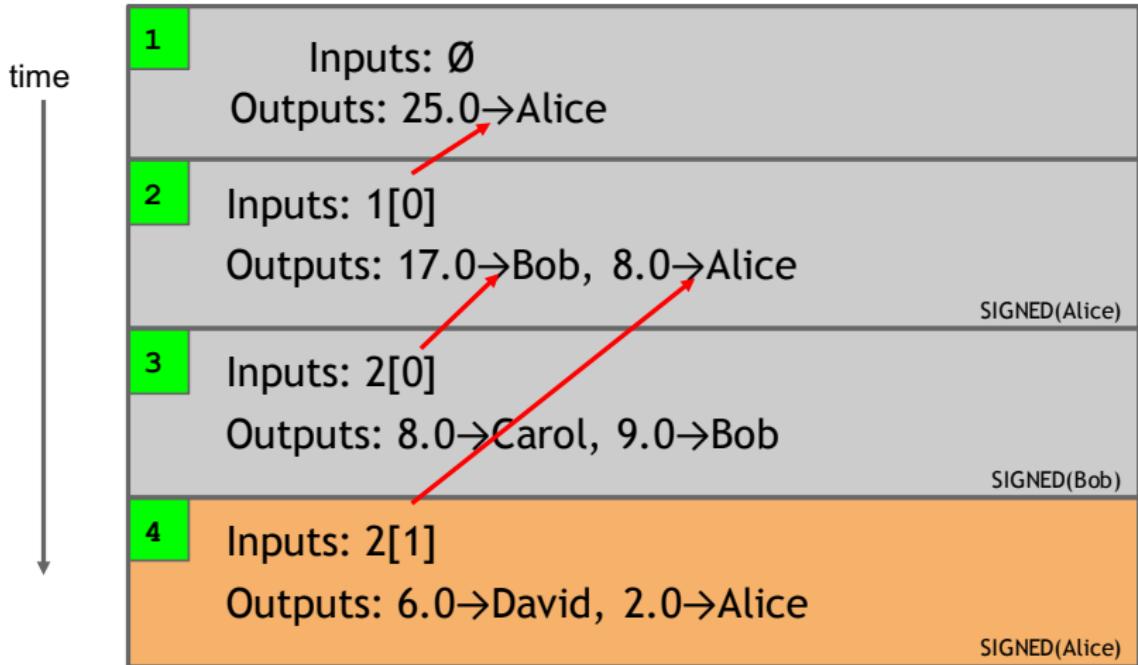
# Chuyển tiền giữa Alice và Bob



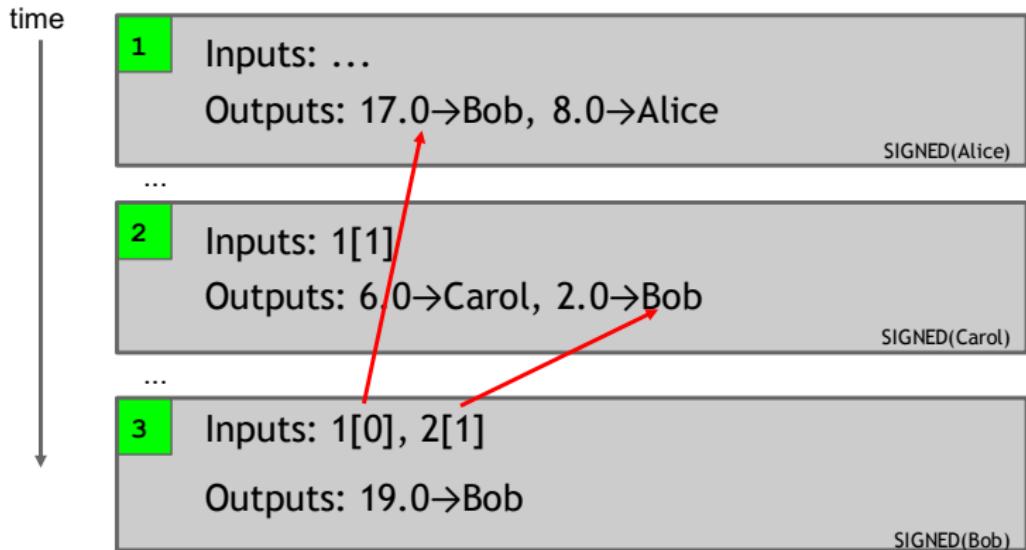


**Hình:** Mạng Bitcoin tạo thêm một Block sau mỗi 10 phút

# Sổ cái dựa trên giao dịch (Bitcoin)

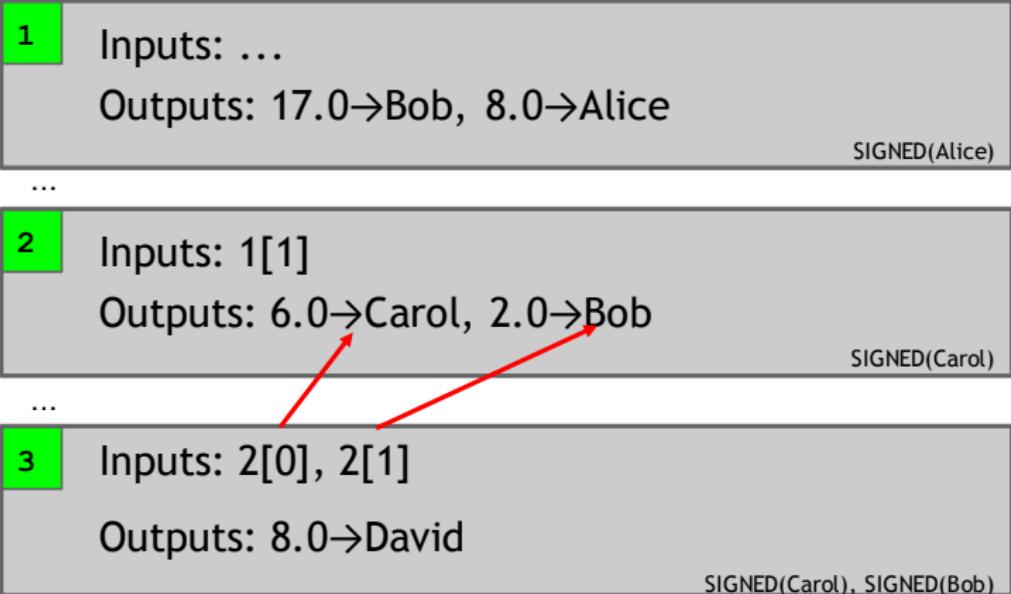


# Gộp nhiều giá trị



# Joint Payment

time



# Giao dịch thực tế

```
{  
    "hash":"5a42590fbe0a90ee8e8747244d6c84f0db1a3a24e8f1b95b10c9e050990b8b6b",  
    "ver":1,  
    "vin_sz":2,  
    "vout_sz":1,  
    "lock_time":0,  
    "size":404,  
    "in": [  
        {  
            "prev_out": [  
                "hash": "3be4ac9728a0823cf5e2deb2e86fc0bd2aa503a91d307b42ba76117d79280260",  
                "n": 0  
            ],  
            "scriptSig": "30440..."  
        },  
        {  
            "prev_out": [  
                "hash": "7508e6ab259b4df0fd5147bab0c949d81473db4518f81afc5c3f52f91ff6b34e",  
                "n": 0  
            ],  
            "scriptSig": "3f3a4ce81..."  
        }  
    ],  
    "out": [  
        {  
            "value": "10.12287097",  
            "scriptPubKey": "OP_DUP OP_HASH160 69e02e18b5705a05dd6b28ed517716c894b3d42e OP_EQUALVERIFY OP_CHECKSIG"  
        }  
    ]  
}
```

## Các thao tác với Bitcoin

- Sinh cặp khóa (hoặc import khóa từ một paper wallet).
- Đưa khóa công khai ("địa chỉ") cho người khác để họ có thể chuyển tiền.
- Tìm khóa công khai của người khác để chuyển tiền.
- Tìm kiếm lịch sử của một địa chỉ.
- Sinh ra một giao dịch Bitcoin để gửi tiền đến một khóa công khai.
- Dùng mã băm của giao dịch như một "biên lai" để tìm kiếm trên mạng.
- Đợi giao dịch được "confirm".

# Nội dung

- ① Giao dịch Bitcoin
- ② Cơ chế đồng thuận của Nakamoto
- ③ Cơ chế thưởng và Bằng chứng công việc
- ④ Đào Bitcoin

# Satoshi Nakamoto

## Bitcoin: A Peer-to-Peer Electronic Cash System

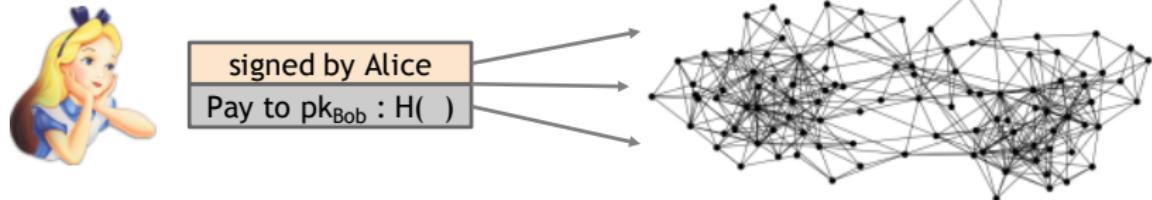
Satoshi Nakamoto  
satoshin@gmx.com  
www.bitcoin.org

**Abstract.** A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

# Cơ chế phi tập trung trong Bitcoin

- Mạng Peer-to-peer: Mở cho mọi người
- Mining (Đào): Mở cho mọi người
- Cập nhật phần mềm: Nhóm phát triển được cộng đồng tin tưởng.

# Bitcoin là hệ thống Peer-to-peer



Khi Alice muốn chuyển tiền cho Bob: Alice phát quảng bá giao dịch tới mọi nút trong mạng

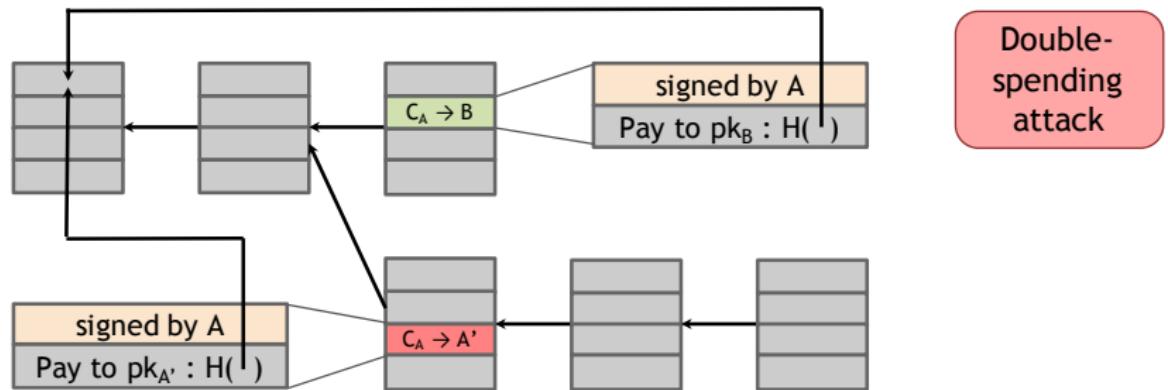
Tại mỗi thời điểm:

- Mọi nút đều có một dãy các Block mà họ đã đồng thuận. Mỗi Block chứa một danh sách các giao dịch.
- Mọi nút có một tập các giao dịch chưa nằm trong blockchain mà họ lắng nghe được.

# Thuật toán đồng thuận của Bitcoin

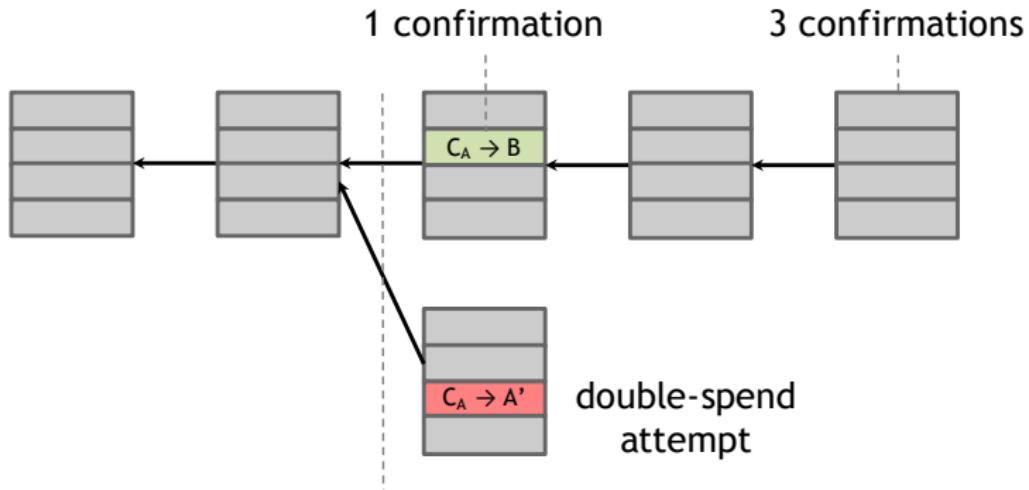
- Các giao dịch mới được phát quảng bá tới mọi nút
- Mỗi nút tập hợp một số giao dịch mới vào trong một Block
- Trong mỗi vòng, một nút **ngẫu nhiên** phải phát quảng bá Block mà nó tạo ra
- Các nút khác chấp nhận Block chỉ nếu mọi giao dịch trong Block này là hợp lệ (chưa được tiêu, chữ ký hợp lệ)
- Các nút thể hiện việc chấp nhận Block này bằng cách thêm mã băm của Block này trong Block tiếp theo mà họ tạo ra.

# Nút không trung thực trong mạng có thể làm gì?



Các nút trung thực trong mạng sẽ mở rộng theo **nhánh hợp lệ dài nhất**.

## Ở góc nhìn của người nhận Bob



Nhìn thấy giao dịch  $C_A \rightarrow B$   
0 confirmations

double-spend  
attempt

- Xác suất bị Double-spending giảm hàm mũ theo số các "confirmation"

Quy tắc chung: 6 confirmation là đủ đảm bảo chắc chắn.

# Tổng kết

- Việc chống giao dịch giả mạo tuy dựa trên Mật mã, nhưng bắt buộc theo cơ chế đồng thuận.
- Việc chống Double-spending thuận túy dựa trên cơ chế đồng thuận.
- Ta không thể chắc chắn 100% một giao dịch nào đó nằm trong nhánh đã được đồng thuận. Chỉ được đảm bảo với một xác suất nào đó.

# Nội dung

- ① Giao dịch Bitcoin
- ② Cơ chế đồng thuận của Nakamoto
- ③ Cơ chế thưởng và Bằng chứng công việc
- ④ Đào Bitcoin

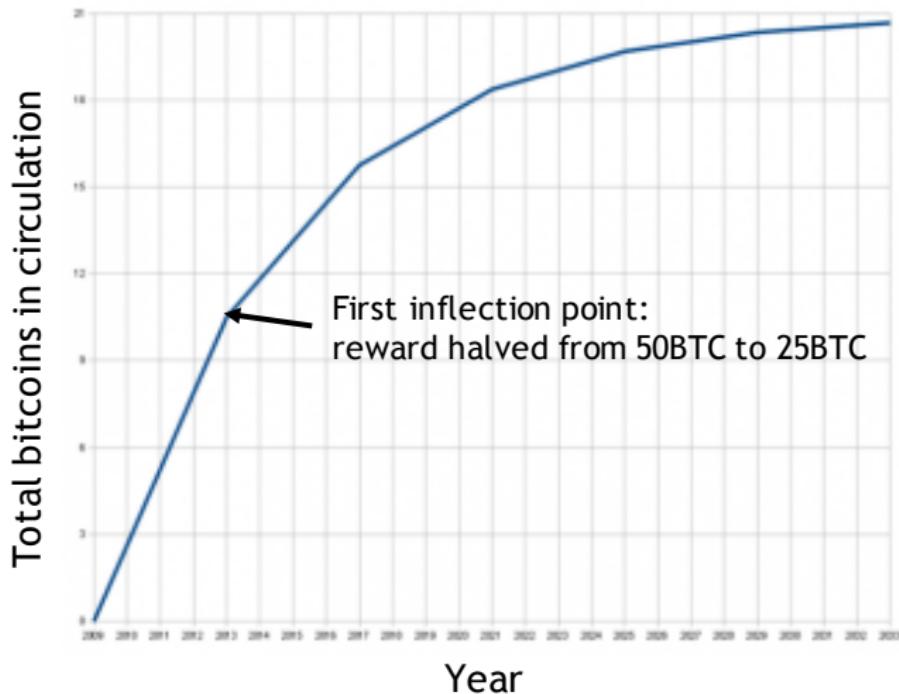
## Thưởng 1: cho việc tạo ra Block mới

Người tạo ra Block mới được

- thêm một *giao dịch tạo coin* vào trong Block
- chọn địa chỉ người nhận cho giao dịch này.

Giao dịch này có giá trị cố định: Hiện tại là 12.5 BTC, giảm một nửa sau mỗi 4 năm.

Người tạo Block lấy được phần thưởng này chỉ nếu Block nằm trong nhánh đồng thuận lâu dài.



- Thưởng cho Block mới là cách tạo ra các bitcoin.
- Sẽ hết vào năm 2040. Sẽ không có thêm bitcoin mới trừ khi thay đổi luật.

## Thưởng 2: Cho phí giao dịch

- Người tạo ra các giao dịch có thể chọn để giá trị output nhỏ hơn giá trị input.
- Phần dư là phí giao dịch và dành cho người tạo ra Block.
- Đây là việc tình nguyện, giống như tiền tip.
- Tuy nhiên người tạo ra Block có thể chọn giao dịch trả phí cao để thêm vào Block và để lại giao dịch có phí thấp.

## Một số vấn đề

- Làm thế nào để chọn **nút ngẫu nhiên**?
- Làm thế nào để tránh bài toán free-for-all?
- Làm thế nào để tránh Sibil attack?

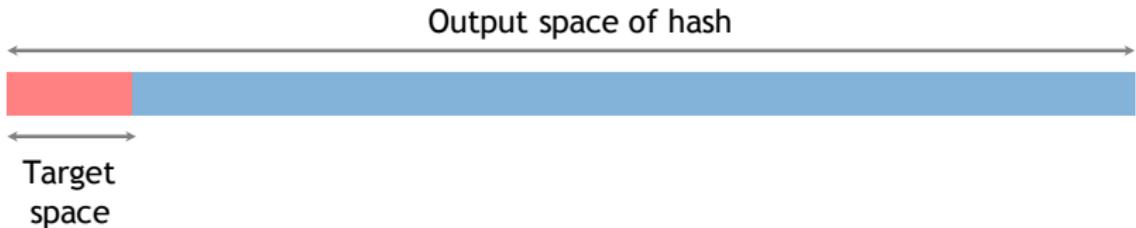
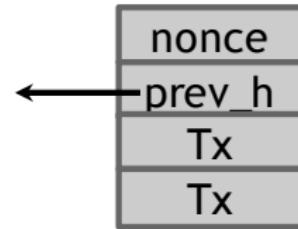
# Bằng chứng công việc (Proof-of-work)

- Để lựa chọn một nút ngẫu nhiên: Ta chọn nút một cách ngẫu nhiên theo tỉ lệ tài nguyên của mỗi nút.  
Ta hy vọng rằng không ai có thể giữ độc quyền tài nguyên này.
- Chọn theo khả năng tính toán: Proof-of-work
- Chọn theo quyền sở hữu: Proof-of-stake

## Hash puzzles

Để tạo ra một Block, ta phải tìm giá trị nonce thỏa mãn

$$H(\text{nonce}|\text{prev\_hash}|\text{tx}| \dots | \text{tx}) < \text{target}.$$



Nếu hàm băm an toàn thì ta chỉ có cách thử các giá trị nonce cho đến khi gặp may.

## Tính chất PoW 1: Rất khó tính toán

- Như tháng 8/2014, khoảng  $10^{20}$  hash/block.
- Chỉ một số nút quan tâm đến việc tính toán này. Đây là các máy đào (Miner).

## Tính chất PoW 2: Tham số hóa chi phí tính toán

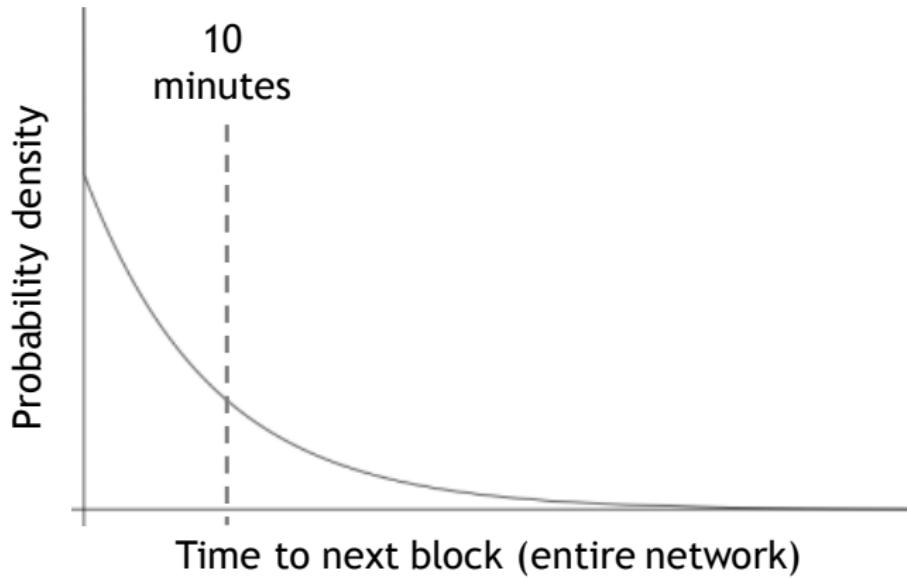
- Các nút tự động tính toán lại target sau 2 tuần.
- Mục đích: Thời gian trung bình để tạo ra block mới là 10 phút.

$\text{Prob}(\text{Alice tạo ra Block tiếp theo}) = \text{tỉ lệ với khả năng hash mà cô ấy có.}$

## Giả sử về tính an toàn

*Không có cách nào tấn công hệ thống nếu phần lớn các Miner (**tính trọng số theo khả năng Hash**) tuân theo giao thức.*

## Giải hash puzzles theo xác suất



Với mỗi miner, thời gian trung bình để tạo được block là

$$\frac{10 \text{ phút}}{\text{tỉ lệ hash mà anh ta có}}$$

## Tính chất PoW 3: Dễ kiểm tra

- nonce được công khai như một phần của block
- Các miners khác chỉ cần kiểm tra

$$H(\text{nonce} \mid \text{prev\_hash} \mid \text{tx} \mid \dots \mid \text{tx}) < \text{target}.$$

## Khi nào thì việc mining có lợi?

*Giá trị thưởng cho việc tạo ra Block mới + phí giao dịch > chi phí cho phần cứng + chi phí điện*

Tuy nhiên việc này rất khó đánh giá vì

- gồm chi phí cố định và chi phí thay đổi
- phần thưởng phụ thuộc vào tốc độ băm tổng thể.

# Nội dung

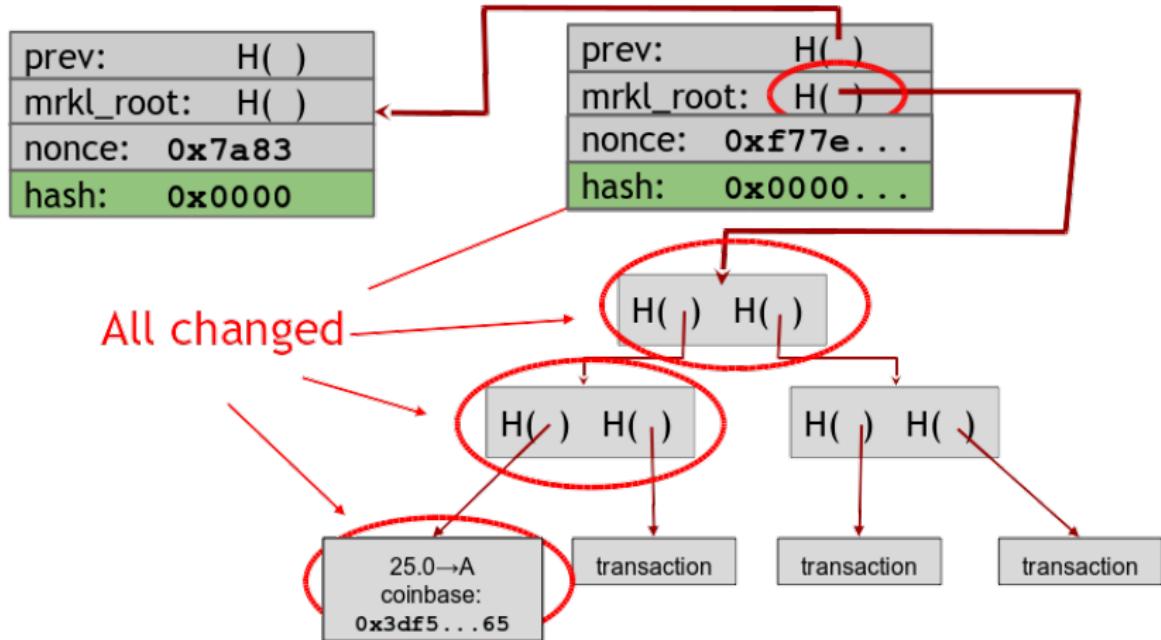
- ① Giao dịch Bitcoin
- ② Cơ chế đồng thuận của Nakamoto
- ③ Cơ chế thưởng và Bằng chứng công việc
- ④ Đào Bitcoin

## 6 bước để đào Bitcoin

- ① Lắng nghe các giao dịch trên mạng và kiểm tra tính hợp lệ của các giao dịch.
- ② Duy trì blockchain và lắng nghe các Block mới. Kiểm tra tính hợp lệ của Block mới được đề xuất.
- ③ Nhóm các giao dịch để đưa vào Block. Phải đảm bảo các giao dịch trong Block là hợp lệ.
- ④ Tìm kiếm giá trị nonce để làm Block của mình hợp lệ.
- ⑤ Hy vọng rằng các miner khác chấp nhận Block của mình.
- ⑥ Profit!

# Tìm Block hợp lệ

Tính cây Merkle và tìm nonce



## 80-byte block header

- 4 bytes: version
- 32 bytes: previous block hash
- 32 bytes: merkle tree of transactions
- 4 bytes: timestamp
- 4 bytes: difficulty target
- 4 bytes: nonce

### Ví dụ

02000000 ..... Block version: 2

b6ff0b1b1680a2862a30ca44d346d9e8

910d334beb48ca0c0000000000000000 ... Hash of previous block's header

9d10aa52ee949386ca9385695f04ede2

70dda20810decd12bc9b048aab31471 ... Merkle root

24d95a54 ..... Unix time: 1415239972

30c31b18 ..... Target: 0x1bc330 \* 256\*\* $(0x18-3)$

fe9f0864 .....Nonce



## Mức độ khó

- Miner phải tìm nonce thỏa mãn

$H(\text{nonce} \mid \text{prev\_hash} \mid \text{tx} \mid \dots \mid \text{tx}) < \text{target}$ .

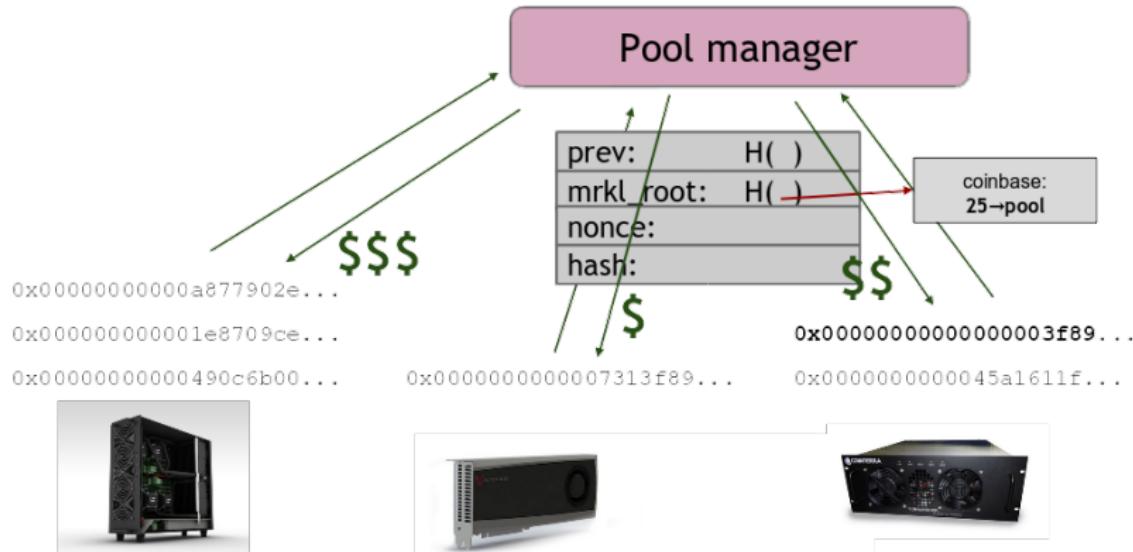


$$\text{next\_difficulty} = \frac{\text{previous\_difficulty} \times \overbrace{2016 \times 10 \text{ phút}}^{2 \text{ tuần}}}{\text{thời gian vừa đào } 2016 \text{ Block}}$$

# CPU Mining

```
while (1){  
    HDR[kNoncePos]++;  
    if (SHA256(SHA256(HDR)) < target)  
        return;  
}
```

# Mining Pool



**Hình:** Nhiều người tham gia đào trên cùng một Block và được trả tiền theo số lượng Hash.

## Một số địa chỉ có ích

- Khóa học Bitcoin and Cryptocurrency Technologies  
<https://www.coursera.org/learn/cryptocurrency>
- Live Blockchain  
<http://blockchain.mit.edu/blockchain/>
- Bitcoin Developer Guide  
<https://bitcoin.org/en/developer-guide>