

# Cryptography Basics for Understanding Bitcoin and Blockchain Technology

---

CHANATHIP NAMPREMPRE, PH.D.

# Agenda

---

## Crypto building blocks

- Hash functions
- Digital signatures

## Relevant data structures

- Hash pointer
- Blockchain

## Simple cryptocurrencies

- Goofycoin
- Scroogecoin

## Bitcoin and blockchain

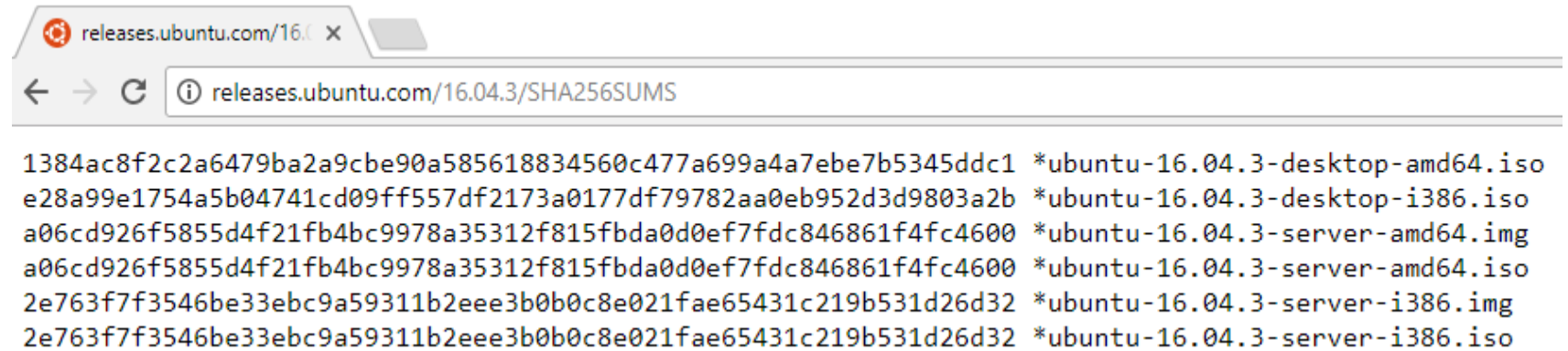
# Hash functions: WHAT

Hash function =  
a function  
mapping inputs  
of any size to  
outputs of a  
fixed size

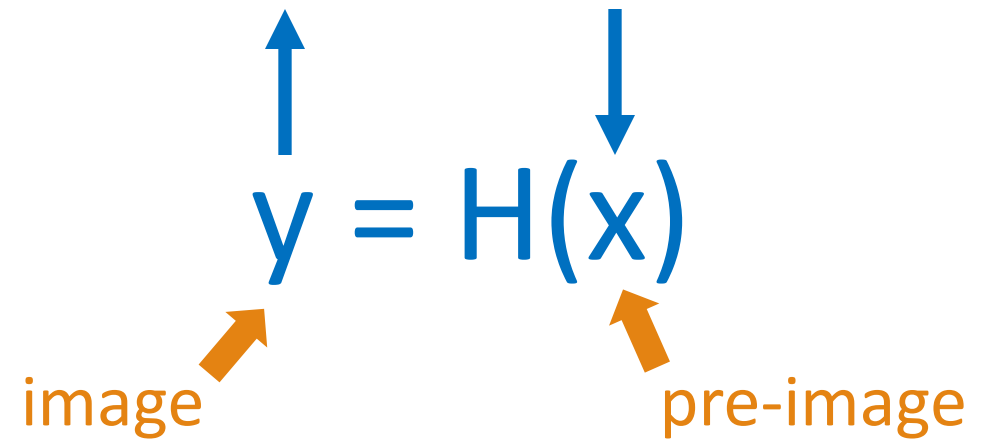
File Name	Date	Time	Size	Description
Parent Directory	-	-	-	-
MD5SUMS	2017-08-03	13:56	395	
MD5SUMS-metalink	2017-08-03	13:16	284	
MD5SUMS-metalink.gpg	2017-08-03	13:16	916	
MD5SUMS.gpg	2017-08-03	13:56	916	
SHA1SUMS	2017-08-03	13:56	443	
SHA1SUMS.gpg	2017-08-03	13:56	916	
SHA256SUMS	2017-08-03	13:56	587	
SHA256SUMS.gpg	2017-08-03	13:56	916	
ubuntu-16.04.3-desktop-amd64.iso	2017-08-01	11:51	1.5G	Desktop image for 64-bit P...
ubuntu-16.04.3-desktop-amd64.iso.torrent	2017-08-03	13:14	59K	Desktop image for 64-bit P...
ubuntu-16.04.3-desktop-amd64.iso.zsync	2017-08-03	13:14	3.0M	Desktop image for 64-bit P...
ubuntu-16.04.3-desktop-amd64.list	2017-08-01	11:51	4.5K	Desktop image for 64-bit P...
ubuntu-16.04.3-desktop-amd64.manifest	2017-08-01	11:45	67K	Desktop image for 64-bit P...
ubuntu-16.04.3-desktop-amd64.metalink	2017-08-03	13:16	47K	Ubuntu 16.04.3 LTS (Xenial
ubuntu-16.04.3-desktop-i386.iso	2017-08-01	11:52	1.5G	Desktop image for 32-bit P...
ubuntu-16.04.3-desktop-i386.iso.torrent	2017-08-03	13:15	60K	Desktop image for 32-bit P...

# Hash functions: WHAT

Hash function =  
a function  
mapping inputs  
of any size to  
outputs of a  
fixed size



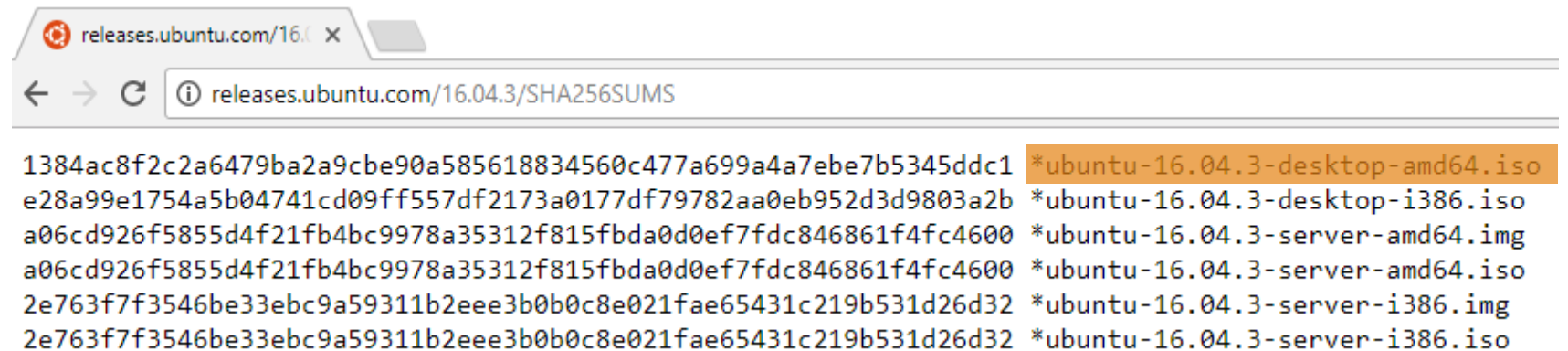
The screenshot shows a browser window with the address bar containing "releases.ubuntu.com/16.04.3/SHA256SUMS". The page content lists several Ubuntu ISO and IMG files with their corresponding SHA256 hashes. The files listed are: \*ubuntu-16.04.3-desktop-amd64.iso, \*ubuntu-16.04.3-desktop-i386.iso, \*ubuntu-16.04.3-server-amd64.img, \*ubuntu-16.04.3-server-amd64.iso, \*ubuntu-16.04.3-server-i386.img, and \*ubuntu-16.04.3-server-i386.iso. Each file name is preceded by a long hexadecimal hash value.



# Hash functions: Collision resistance

---

Hard to find  $x \neq y$  such that  
 $H(x) = H(y)$



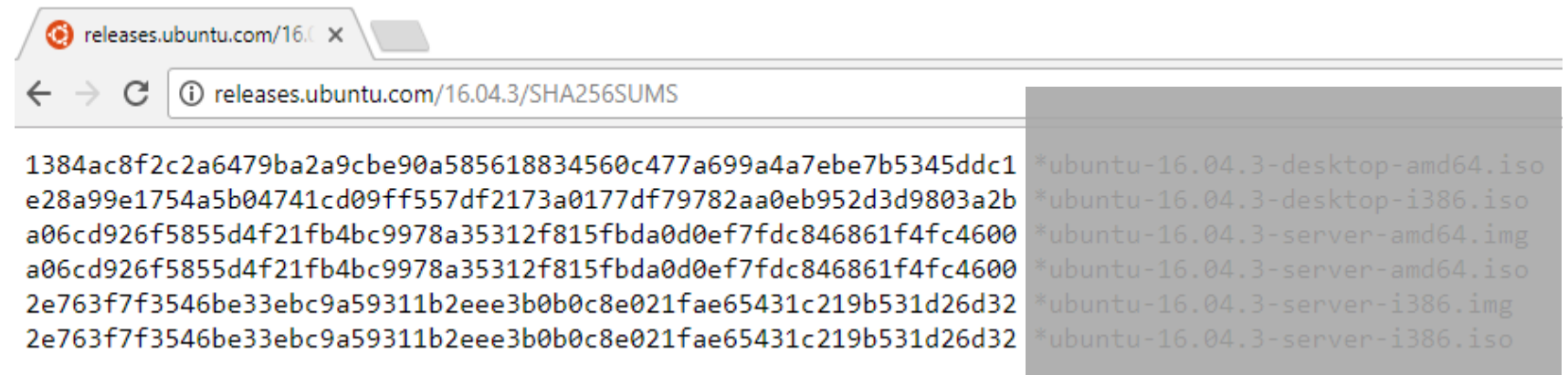
```
releases.ubuntu.com/16.04.3/SHA256SUMS
1384ac8f2c2a6479ba2a9cbe90a585618834560c477a699a4a7ebe7b5345ddc1 *ubuntu-16.04.3-desktop-amd64.iso
e28a99e1754a5b04741cd09ff557df2173a0177df79782aa0eb952d3d9803a2b *ubuntu-16.04.3-desktop-i386.iso
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.img
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.iso
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.img
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.iso
```

# Hash functions: Hiding

---

If a secret value  $r$  is sampled from a probability distribution that has high min-entropy, then

given  $y = H(r \parallel x)$ , it is hard to find  $x$ .



The screenshot shows a web browser window with the address bar containing `releases.ubuntu.com/16.04.3/SHA256SUMS`. The page content displays a list of SHA256 hashes for various Ubuntu 16.04.3 ISO and IMG files. The hashes are arranged in two columns, with the first column containing the full hash and the second column containing the filename. The files listed include desktop and server versions for both amd64 and i386 architectures.

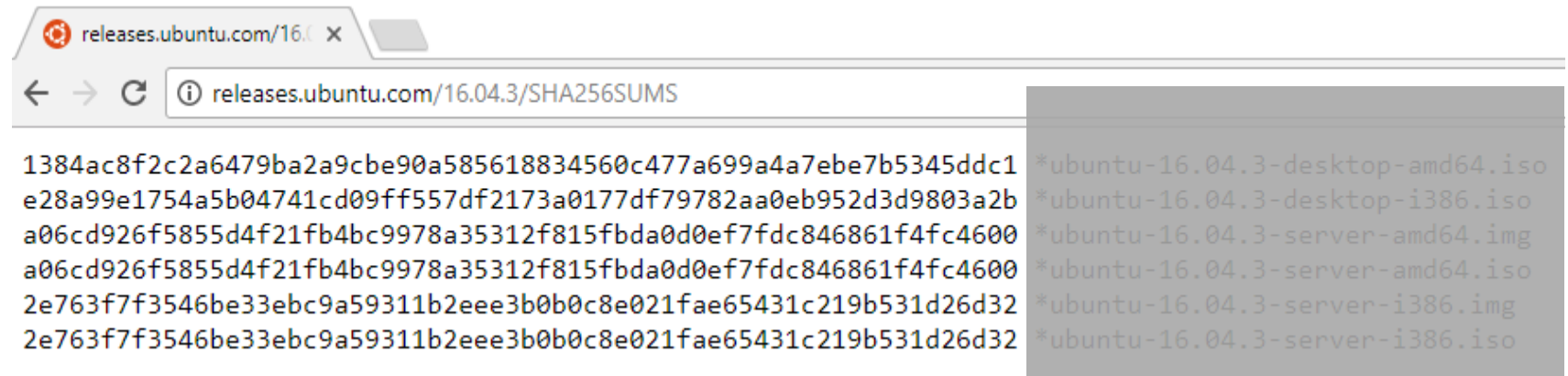
```
1384ac8f2c2a6479ba2a9cbe90a585618834560c477a699a4a7ebe7b5345ddc1 *ubuntu-16.04.3-desktop-amd64.iso
e28a99e1754a5b04741cd09ff557df2173a0177df79782aa0eb952d3d9803a2b *ubuntu-16.04.3-desktop-i386.iso
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.img
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.iso
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.img
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.iso
```

# Hash functions: Puzzle friendliness

---

If a secret value  $r$  is chosen from a probability distribution that has high min-entropy, then **for every  $n$ -bit output  $y$**

given  $y = H(r \parallel x)$ , it is hard to find  $x$  **in time much less than  $2^n$ .**

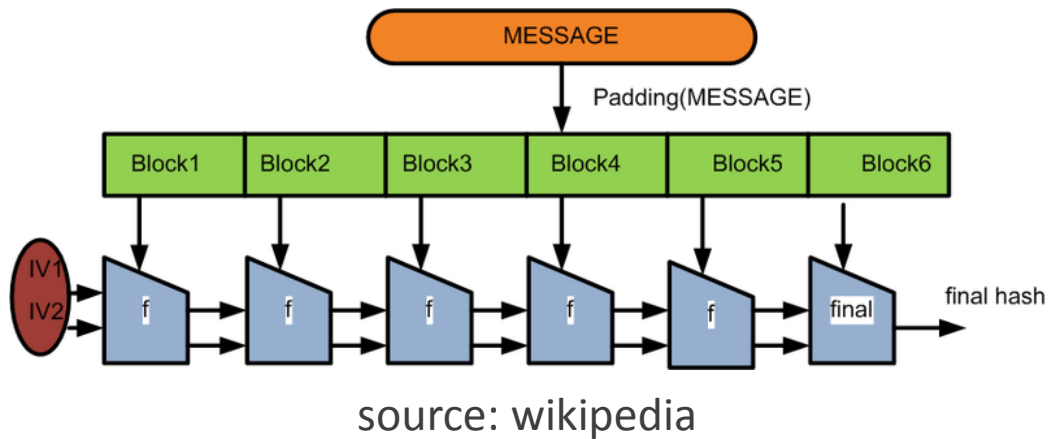


The screenshot shows a web browser window with the address bar containing `releases.ubuntu.com/16.04.3/SHA256SUMS`. The page content displays a list of SHA256 hashes for various Ubuntu 16.04.3 release images, with the first two lines of the list highlighted in a grey box.

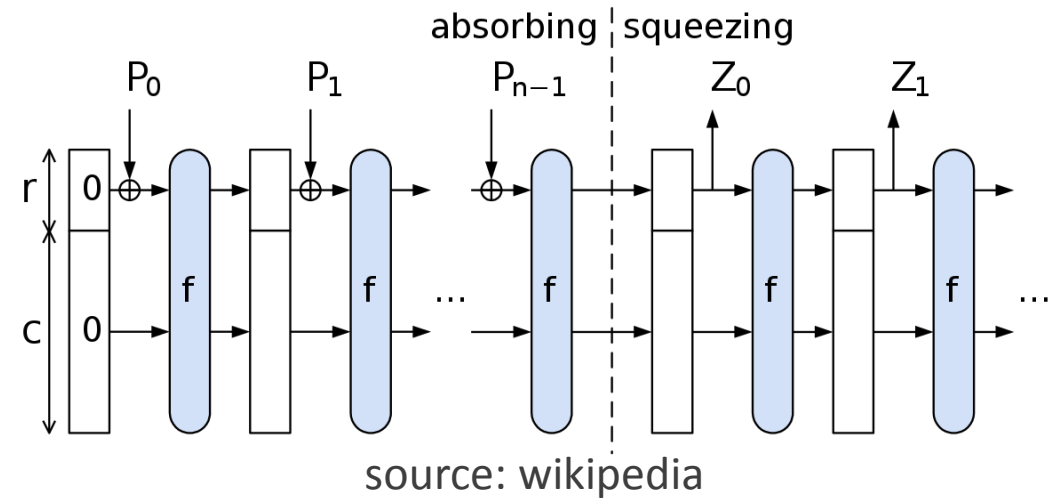
```
1384ac8f2c2a6479ba2a9cbe90a585618834560c477a699a4a7ebe7b5345ddc1 *ubuntu-16.04.3-desktop-amd64.iso
e28a99e1754a5b04741cd09ff557df2173a0177df79782aa0eb952d3d9803a2b *ubuntu-16.04.3-desktop-i386.iso
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.img
a06cd926f5855d4f21fb4bc9978a35312f815fbda0d0ef7fdc846861f4fc4600 *ubuntu-16.04.3-server-amd64.iso
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.img
2e763f7f3546be33ebc9a59311b2eee3b0b0c8e021fae65431c219b531d26d32 *ubuntu-16.04.3-server-i386.iso
```

# Hash functions: Examples

- SHA256



- Keccak



- Chacha
- Curl (!)
- etc



# Digital signatures: WHAT

---

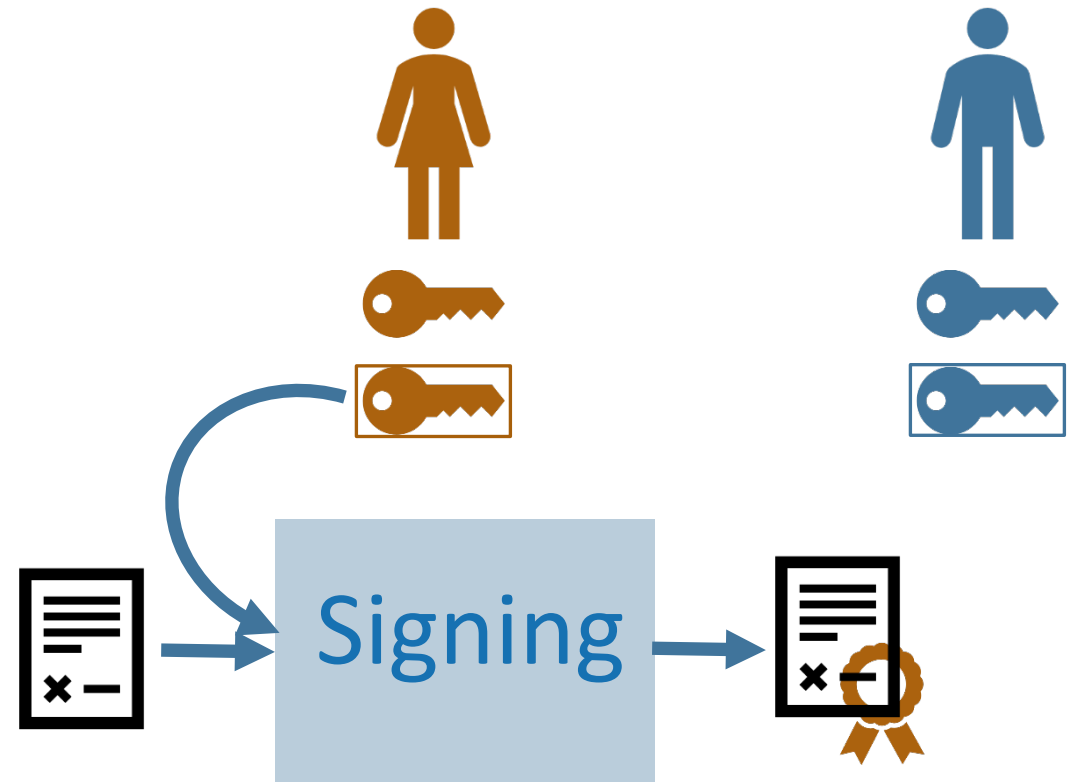
## Key generation:

everyone in the system has a key pair

(public key, private key)

**Signing:** only the signer can sign (duh!)

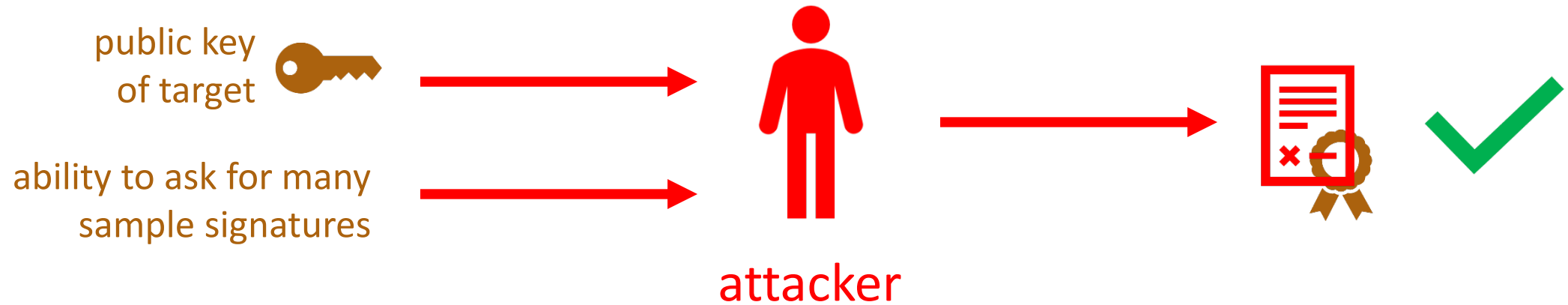
**Verification:** “anyone” can verify



# Digital signatures: unforgeability

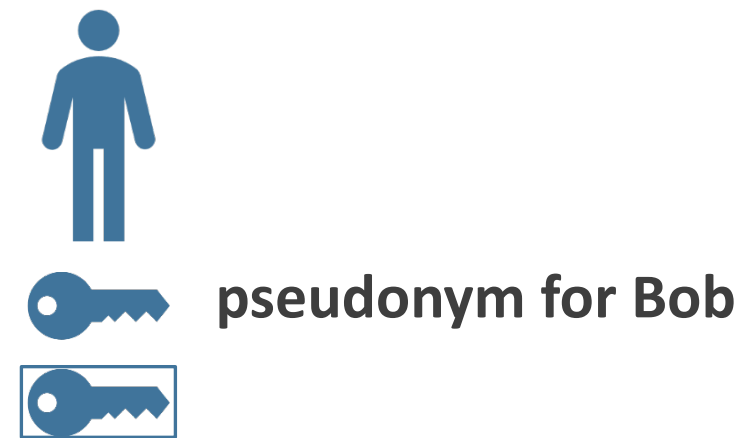
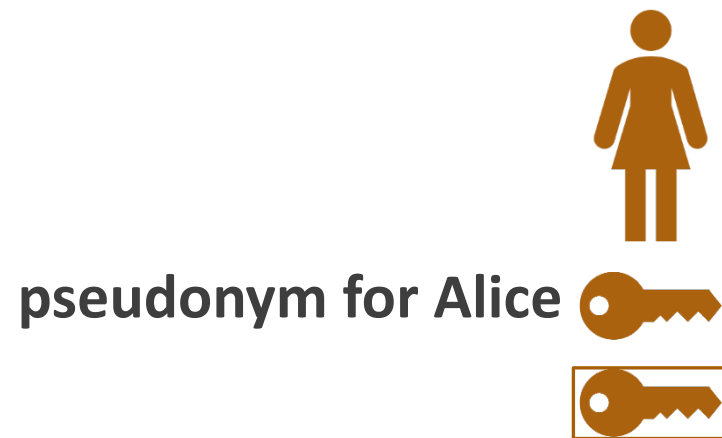
---

Hard to forge signatures



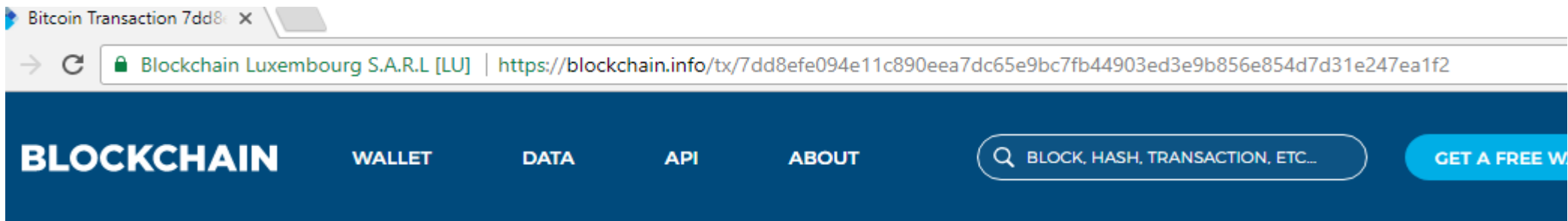
# Digital signatures: Public keys as identities

---



# Public key as identities in bitcoin system

<https://blockchain.info/tx/7dd8efe094e11c890eea7dc65e9bc7fb44903ed3e9b856e854d7d31e247ea1f2>



## Transaction View information about a bitcoin transaction

7dd8efe094e11c890eea7dc65e9bc7fb44903ed3e9b856e854d7d31e247ea1f2

1HhZFDi1zVRpj1fKi3ACH6Y83XBFuKsJ2k  
1AqTMY7kmHZxBuLUR5wJjPFUvqGs23sesr  
19iD73ub6VLM671kUi8WaWfDuz86tYobiH



1EF2z1RgvEf2WEqDZo7A4UeACqghJAxbrW  
1HiJ5WUNtKM6iRLbDPHqpxjBEVKzKHSF1a  
1EPq2pgn11bi1H24mEgEcKuNunFTbSKgCS  
18J9rgofxkwHE1FMV17io9k5kRonNZczZ3  
1JZeGW7EoKQUEbtRhtXg7dtpeCJ1NGXoKh  
1KnfCxy8XkNQu9pYpqEU1fxAELqRsYDbZK  
1FSUfocsChwPz6RK4E6sKDbxJ1Ehw73BFo  
12jZdwu4nnzUEL7Xn5em79adxV3iaE33V7  
12dZBgBEz5yU5z5ko7WjCd9ceuexkXHy1E  
16pcddk8c6MFKyQvL2NvuWEguGw1R8B3VL  
18leYvf1Bsr2Ax8b2It9erbS71qiHPWPan

0.59111937 BTC  
0.8731894 BTC  
1.01440538 BTC  
1.47501259 BTC  
1.04457412 BTC  
2.21591053 BTC  
0.58086517 BTC  
1.01820914 BTC  
1.0134059 BTC  
1.00749482 BTC  
1.01820272 BTC

# Digital signatures: Examples

---

- RSA
  - ElGamal
  - DSA
  - etc.
- 
- Bitcoin system uses ECDSA.

# DSA (Digital Signature Algorithm)

- Fix primes  $p, q$  such that  $q$  divides  $p - 1$
- Let  $G = \mathbf{Z}_p^* = \langle h \rangle$  and  $g = h^{(p-1)/q}$  so that  $g \in G$  has order  $q$
- $H: \{0, 1\}^* \rightarrow \mathbf{Z}_q$  a hash function
- Signer keys:  $pk = X = g^x \in \mathbf{Z}_p^*$  and  $sk = x \stackrel{\$}{\leftarrow} \mathbf{Z}_q$

## Alg $S_x(M)$

$m \leftarrow H(M)$

$k \stackrel{\$}{\leftarrow} \mathbf{Z}_q^*$

$r \leftarrow (g^k \bmod p) \bmod q$

$s \leftarrow (m + xr) \cdot k^{-1} \bmod q$

Return  $(r, s)$

## Alg $V_X(M, (r, s))$

$m \leftarrow H(M)$

$w \leftarrow s^{-1} \bmod q$

$u_1 \leftarrow mw \bmod q$

$u_2 \leftarrow rw \bmod q$

$v \leftarrow (g^{u_1} X^{u_2} \bmod p) \bmod q$

If  $(v = r)$  then return 1 else return 0

source: Bellare  
lecture slides.

Details: Signature is regenerated if  $s = 0$ .

# Agenda

---

## Crypto building blocks

- Hash functions
- Digital signatures

## Relevant data structures

- Hash pointer
- Blockchain

## Simple cryptocurrencies

- Goofycoin
- Scroogecoin

## Bitcoin and blockchain

# Relevant data structures

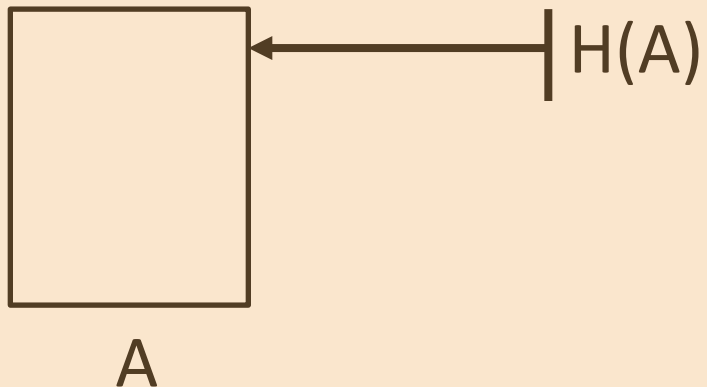
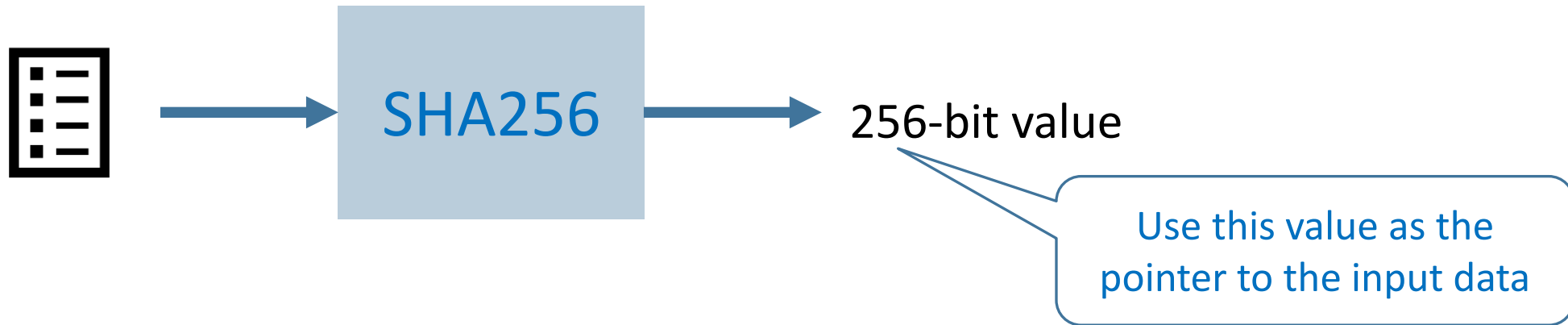
---

Hash pointer

Blockchain



# Hash pointer

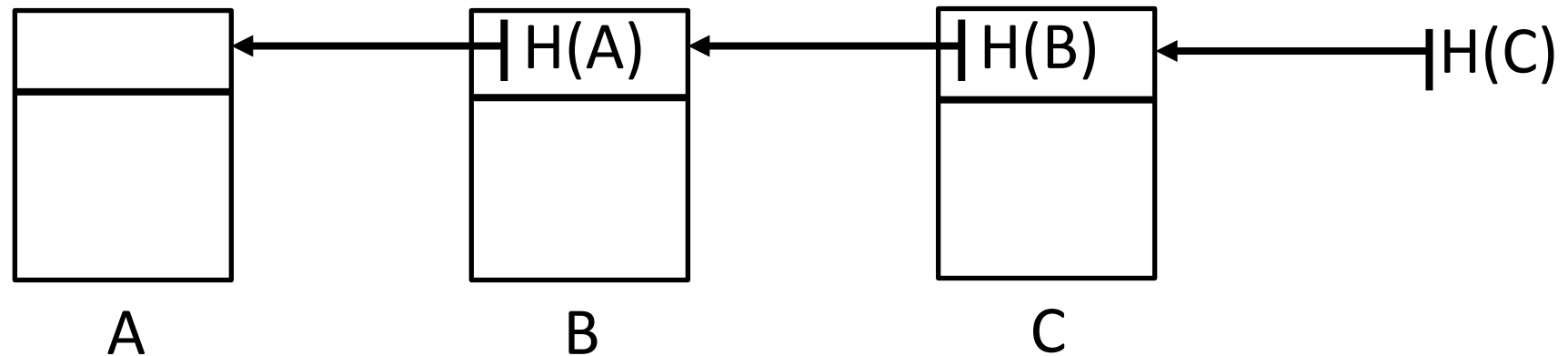


## Notation for Hash pointer

Property: Lets us locate the data and verify the contents.

# Blockchain (as a data structure)

---

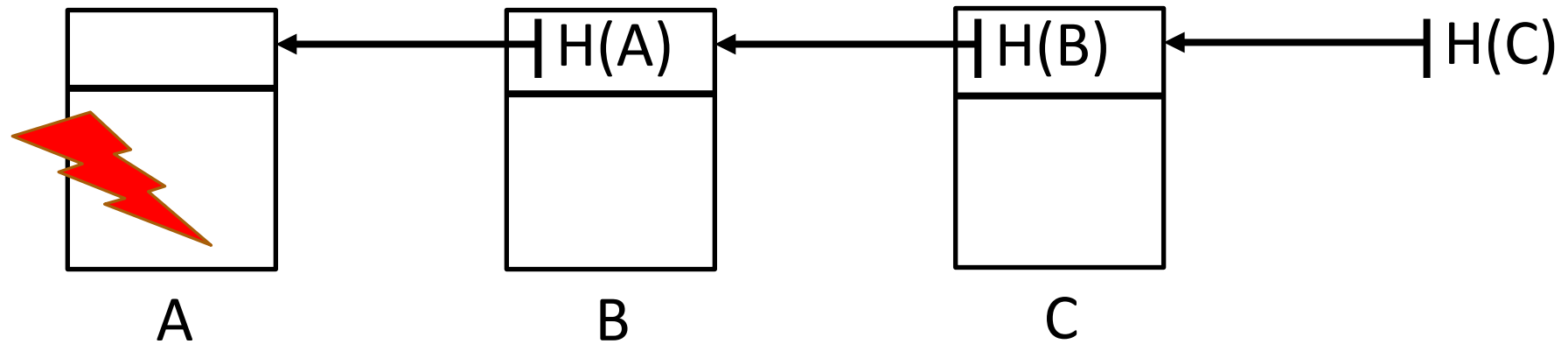


In its simplest form, a blockchain is a linked list created by embedding hash pointers into the data blocks.

Verify data in the entire chain by verifying the hashes in the chain.

# Blockchain (as a data structure)

---



- What if an attacker wants to modify data in block A?
- Compared to a normal, line-by-line ledger?

Example applications: append-only ledger, “timestamping” service

# Agenda

---

## Crypto building blocks

- Hash functions
- Digital signatures

## Relevant data structures

- Hash pointer
- Blockchain

## Simple cryptocurrencies

- Goofycoin
- Scroogecoin

## Bitcoin and blockchain

# Simple cryptocurrencies

---

Goofycoin

Scroogecoin

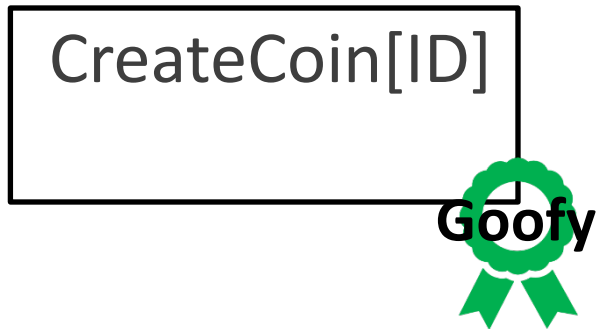
Reference:

*Bitcoin and Cryptocurrency Technologies*, Narayanan, et. al. 2016

# Goofycoin

---

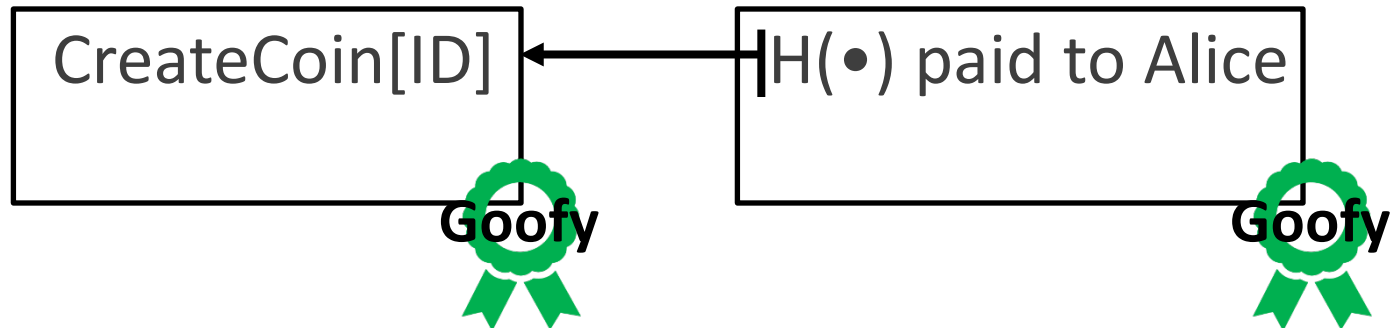
Goofy is the central authority. He mints coins.



# Goofycoin

---

Goofy is the central authority. He pays people.

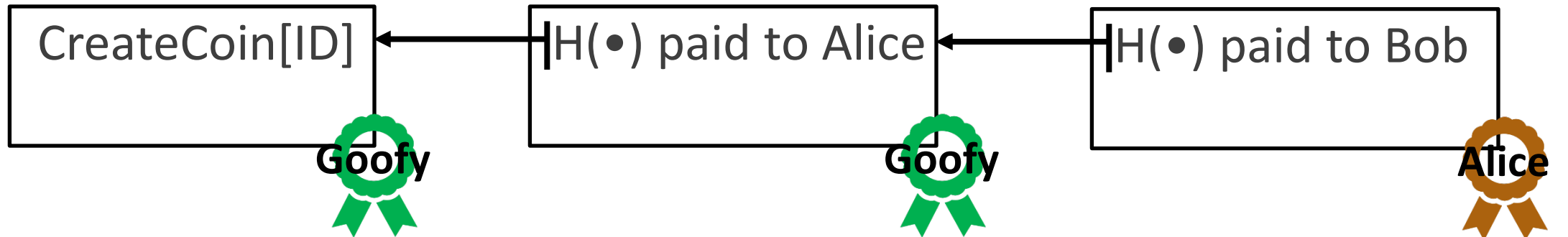


Simplification: one transaction per block.

# Goofycoin

---

People spend coins.

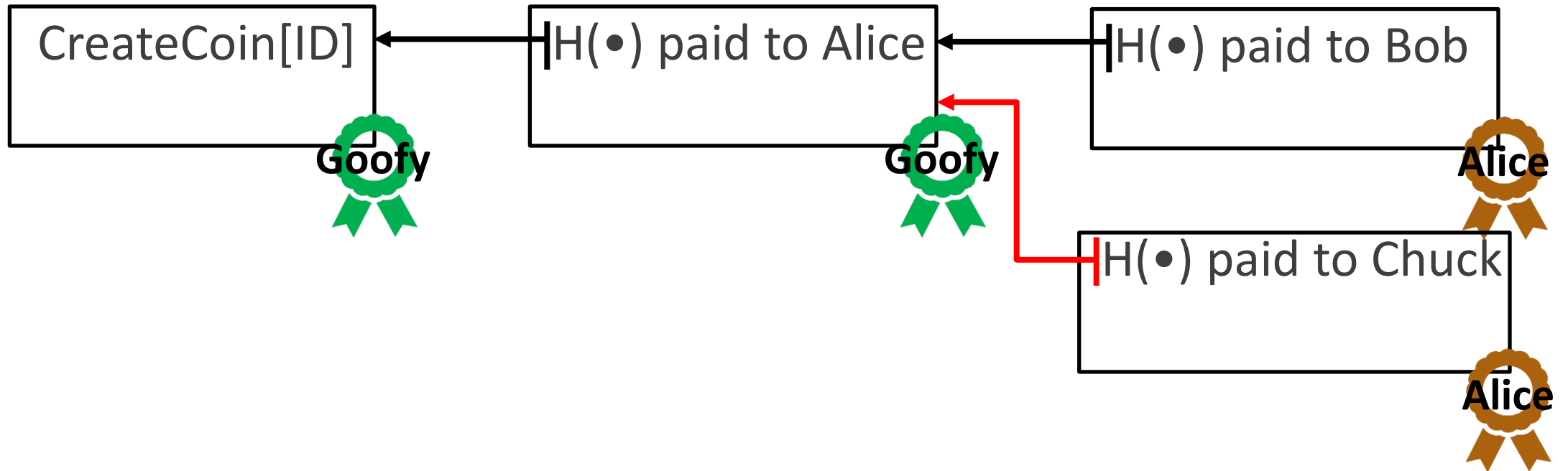




# Goofycoin

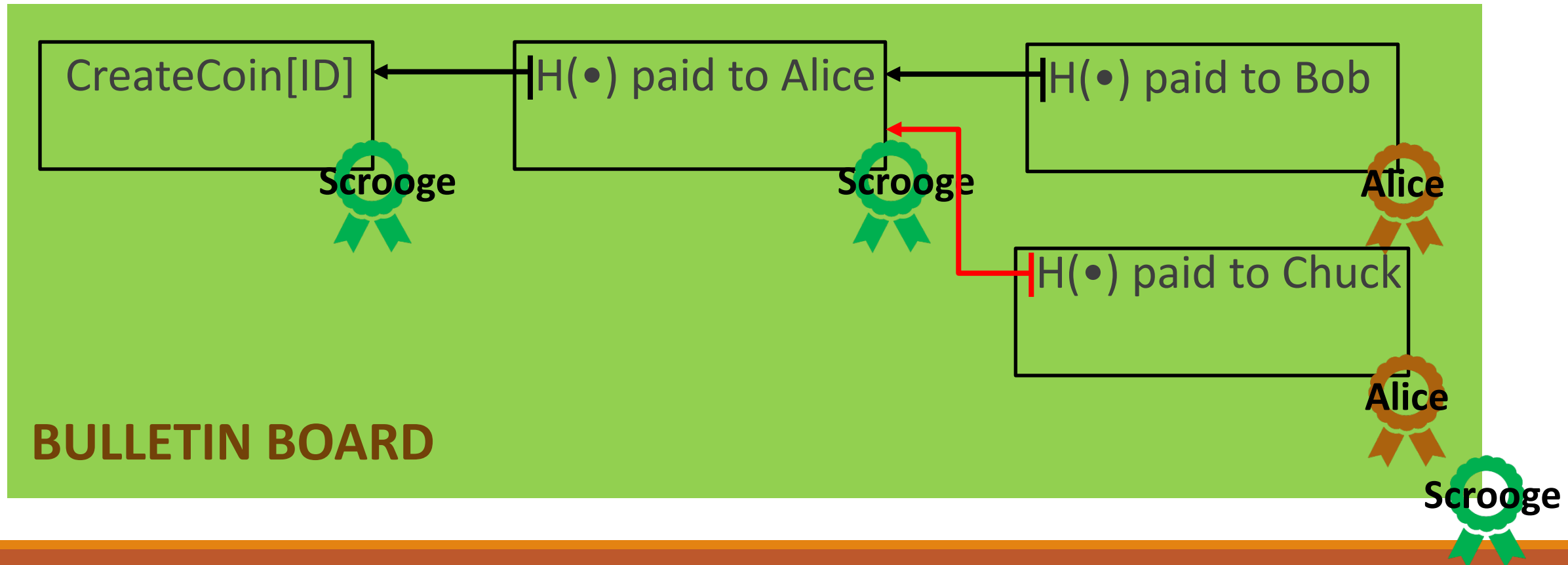
---

But what if Alice tries to **double-spend**?



# Scroogecoin

Scrooge acts like Goofy except that he publishes the history of all transactions.



# Scroogecoin

---

Scrooge is the central authority.

Reasons we want to remove him:

- We don't necessarily trust him
- He's the single point of failure.
  - if he decides to quit this job or
  - if his secret key gets stolen or
  - if his computer crashes
  - etc etc, we're toast!

# Agenda

---

## Crypto building blocks

- Hash functions
- Digital signatures

## Relevant data structures

- Hash pointer
- Blockchain

## Simple cryptocurrencies

- Goofycoin
- Scroogecoin

## Bitcoin and blockchain

# Bitcoin and blockchain

---

Bitcoin network

Bitcoin blocks

Adding blocks to the blockchain

Bitcoin transactions

Attacks

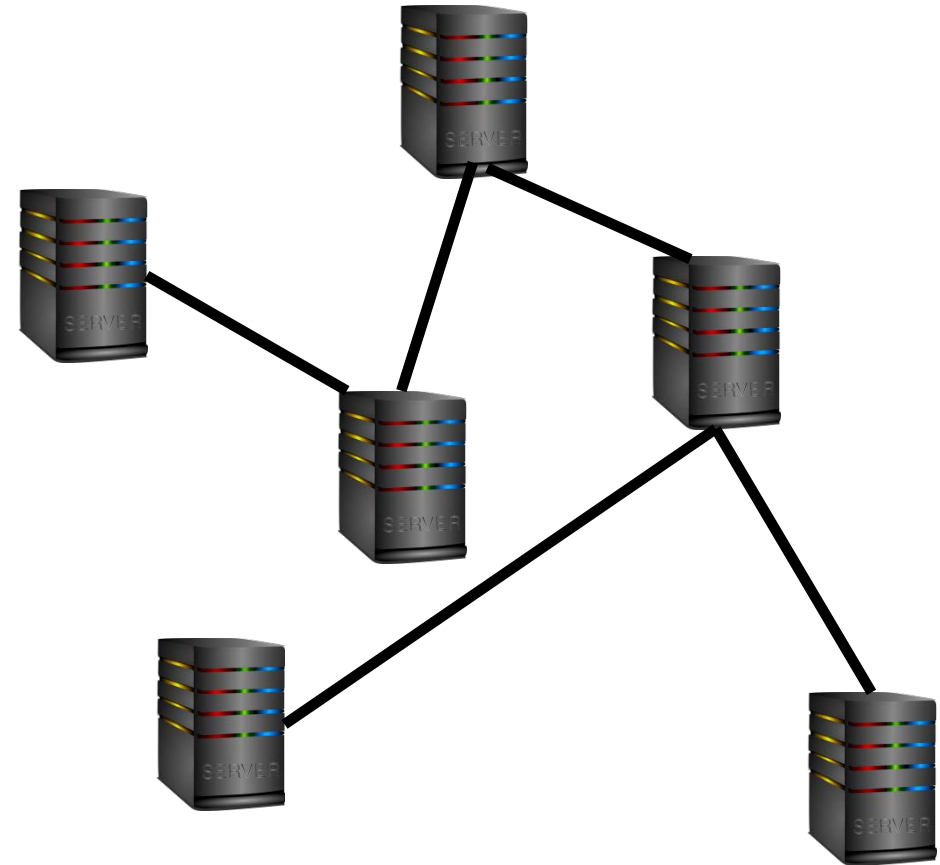
- Fraudulent transactions
- Double spending attack

Discussions

# Bitcoin network

---

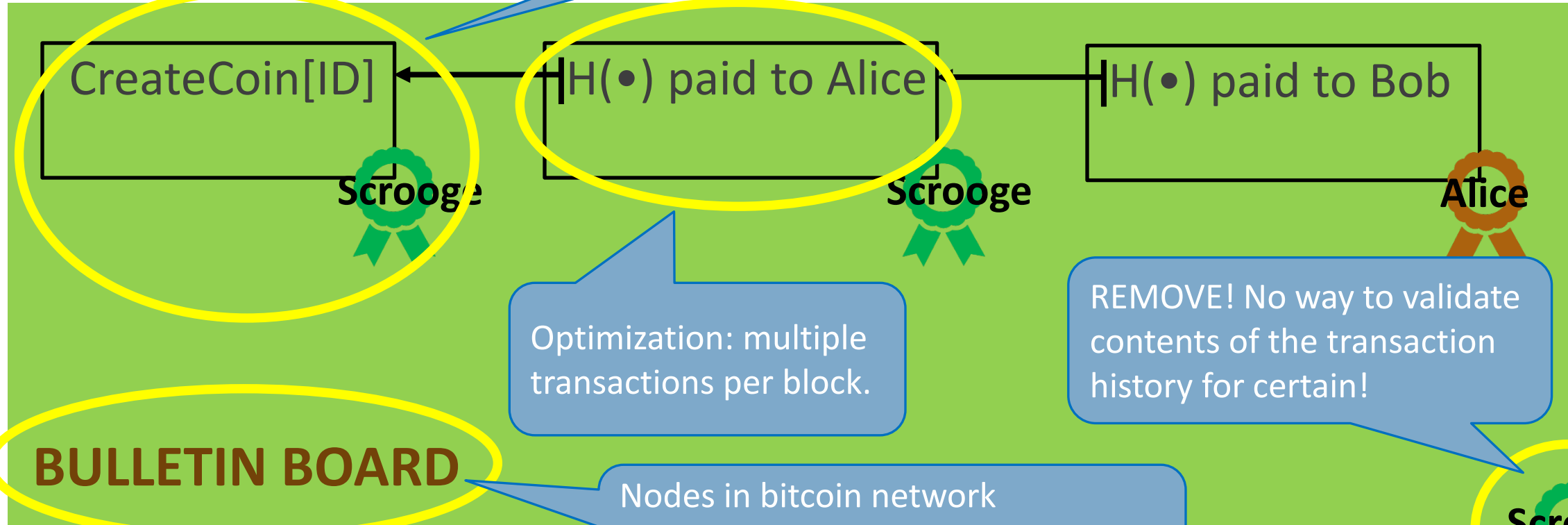
- Peer-to-peer
- No trusted authority
- Anyone can run a node
- Nodes execute distributed protocols
- Nodes potentially malicious



# Getting to Bitcoin

Recall Scroogecoin.

Coins are created as mining reward only.



# Bitcoin blockchain

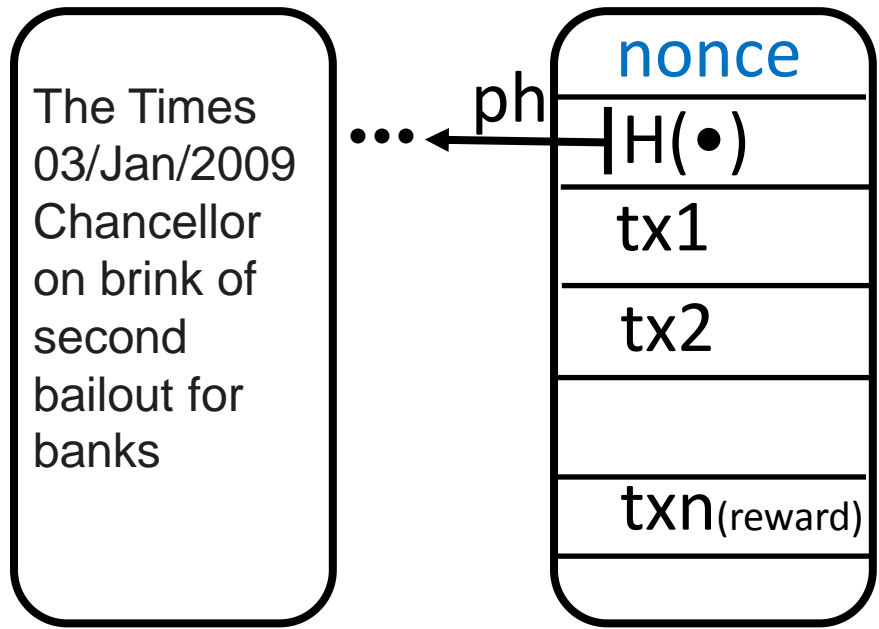
The Times  
03/Jan/2009  
Chancellor  
on brink of  
second  
bailout for  
banks

Genesis  
block





# Bitcoin blockchain: appending a new block



Genesis block

Each **bitcoin node**

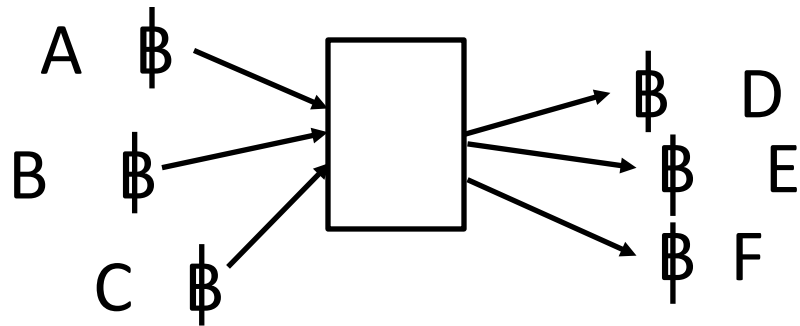
1. collects and **validates** transactions conducted by bitcoin users
2. tries to add the next block onto the blockchain to reap the **block reward**

halved every 4 years

To add the next block, find “**nonce**” such that  $00\dots0xxxxx\dotsxxxx = H(\text{nonce}||\text{ph}||\text{tx1}||\text{tx2} \dots)$

re-configured every 2 weeks

# Bitcoin transaction: money changing hands



Coins are **immutable**. (can only be created and consumed)

A transaction is valid if

- consumed coins were created in previous transaction
- total value out  $\leq$  total value in, and
- entire transaction signed by owners of all consumed coins
- consumed coins not a double-spend

← no guarantee!

transID: 73		type:PayCoins	
consumed coinIDs: 68(1), 42(0), 72(3)			
coins created			
<i>num</i>	<i>value</i>	<i>recipient</i>	
0	3.2	0x...	
1	1.4	0x...	
2	7.1	0x...	
signatures			

source: Bitcoin and Cryptocurrency Technologies, Narayanan, et. al. 2016

# Bitcoin transaction

transID: 73    type:PayCoins		
consumed coinIDs: 68(1), 42(0), 72(3)		
coins created		
<i>num</i>	<i>value</i>	<i>recipient</i>
0	3.2	0x...
1	1.4	0x...
2	7.1	0x...
signatures		

source: Bitcoin and Cryptocurrency Technologies, Narayanan, et. al. 2016

Transaction ID

in:

←  $H(\bullet)$ , index<sub>1</sub>, signature<sub>1</sub>, pk<sub>s1</sub>

←  $H(\bullet)$ , index<sub>2</sub>, signature<sub>2</sub>, pk<sub>s2</sub>

←  $H(\bullet)$ , index<sub>3</sub>, signature<sub>3</sub>, pk<sub>s3</sub>

out:

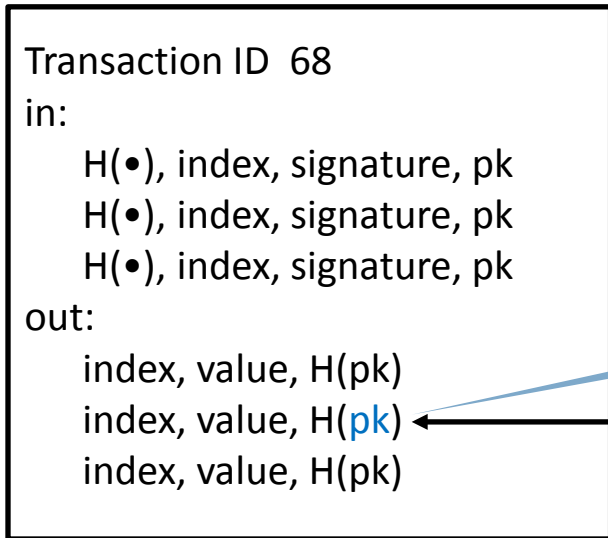
index<sub>1</sub>, value<sub>1</sub>, H(pk<sub>r1</sub>)

index<sub>2</sub>, value<sub>2</sub>, H(pk<sub>r2</sub>)

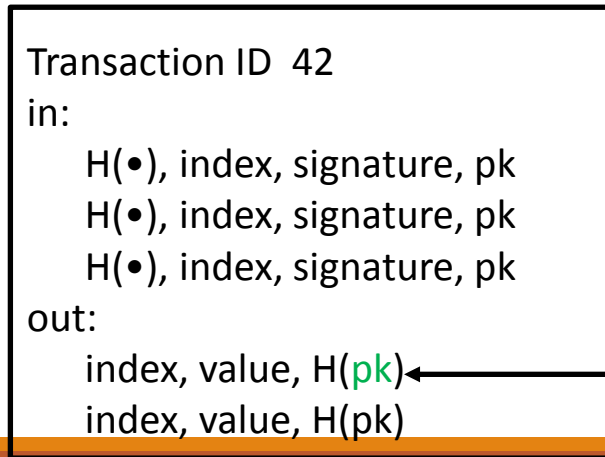
index<sub>3</sub>, value<sub>3</sub>, H(pk<sub>r3</sub>)

# Bitcoin transaction

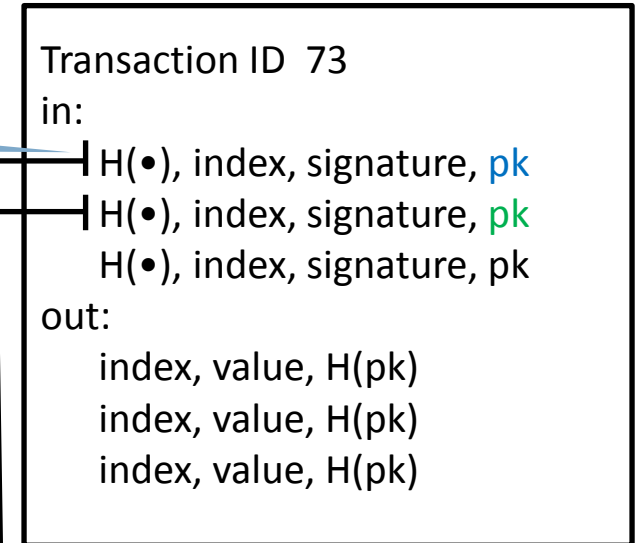
stack-based language



referenced transaction



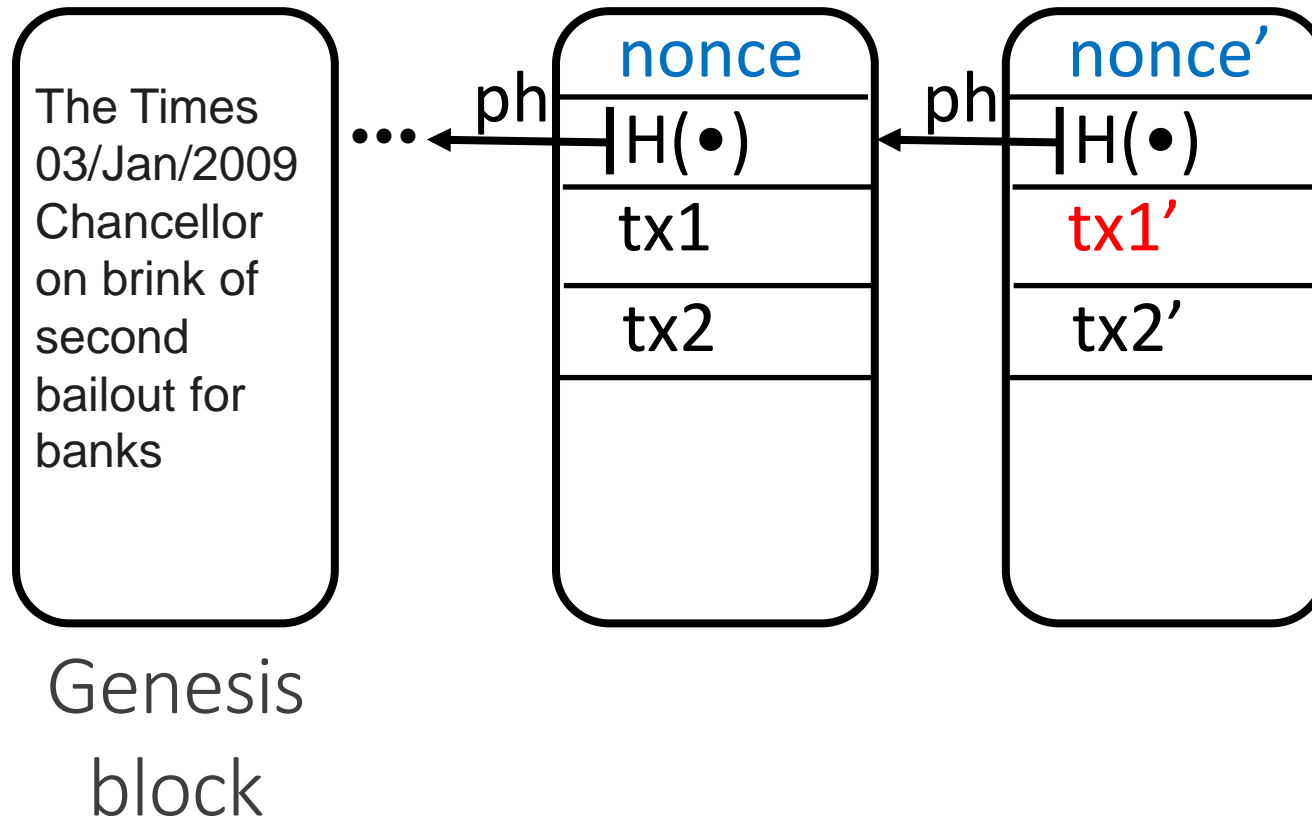
referenced transaction



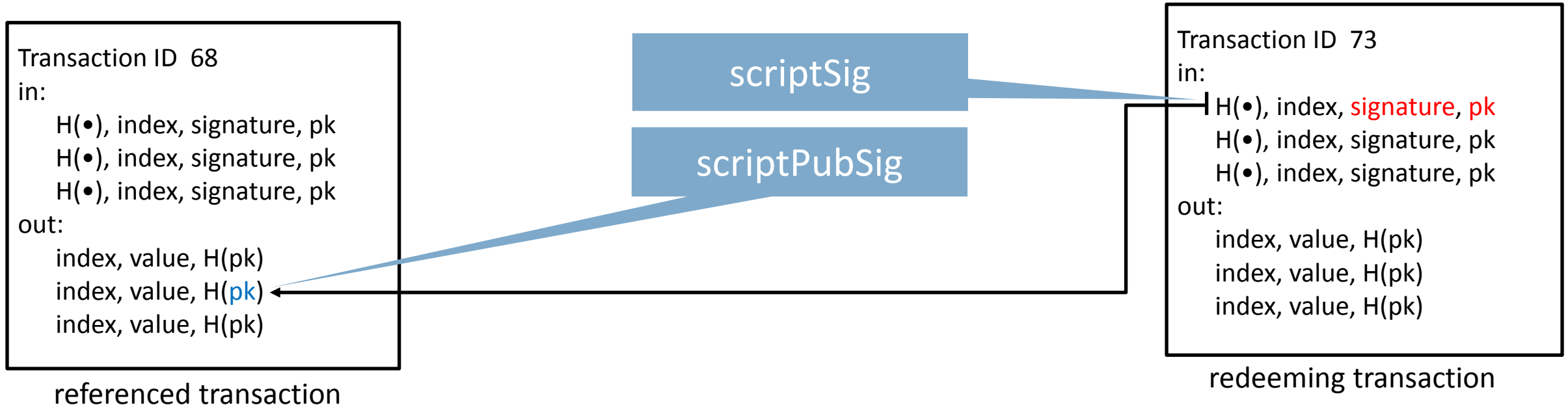
redeeming transaction

# Fraudulent transactions?

---



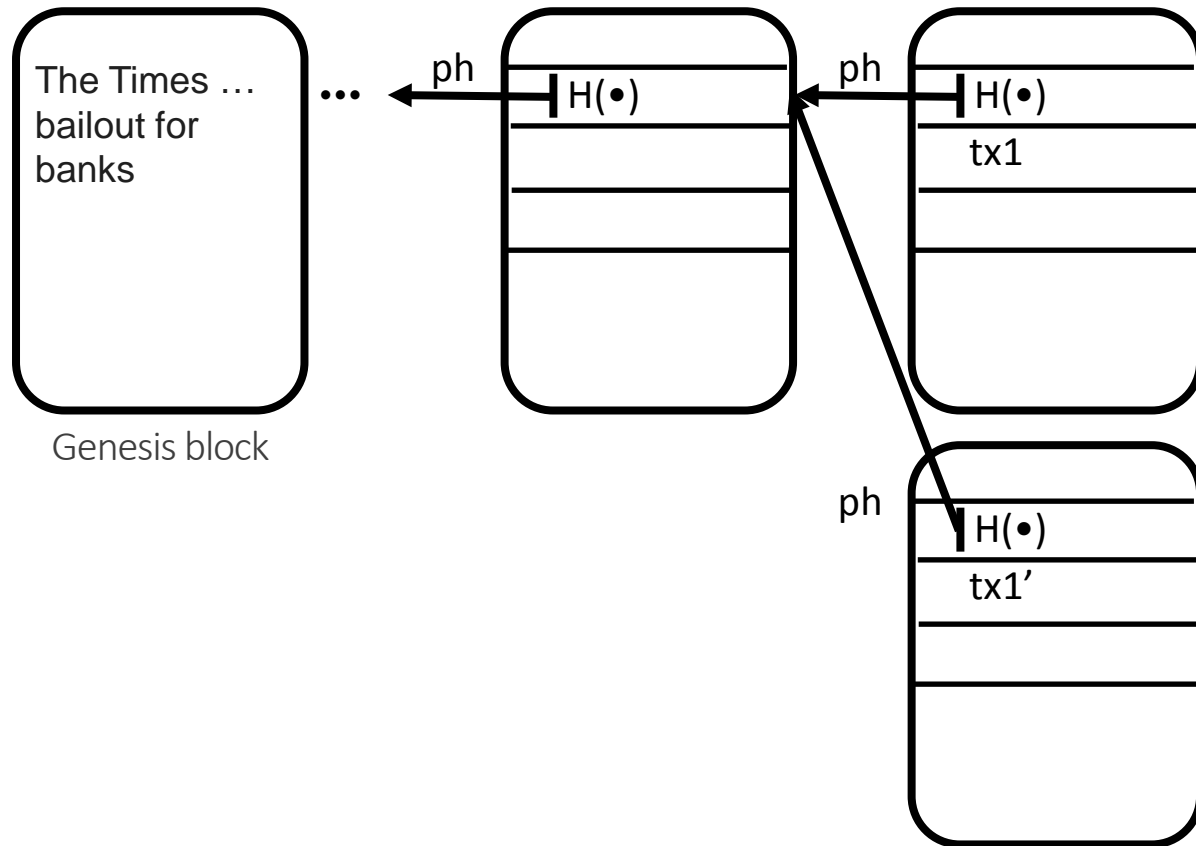
# Fraudulent transaction?



Prevented through **cryptology** and enforced through **consensus**.

- Bitcoin nodes will detect fraudulent transactions when verifying signatures.
- If majority of nodes are honest, the blocks with invalid transactions will not be added to the blockchain.

# Double-spending?

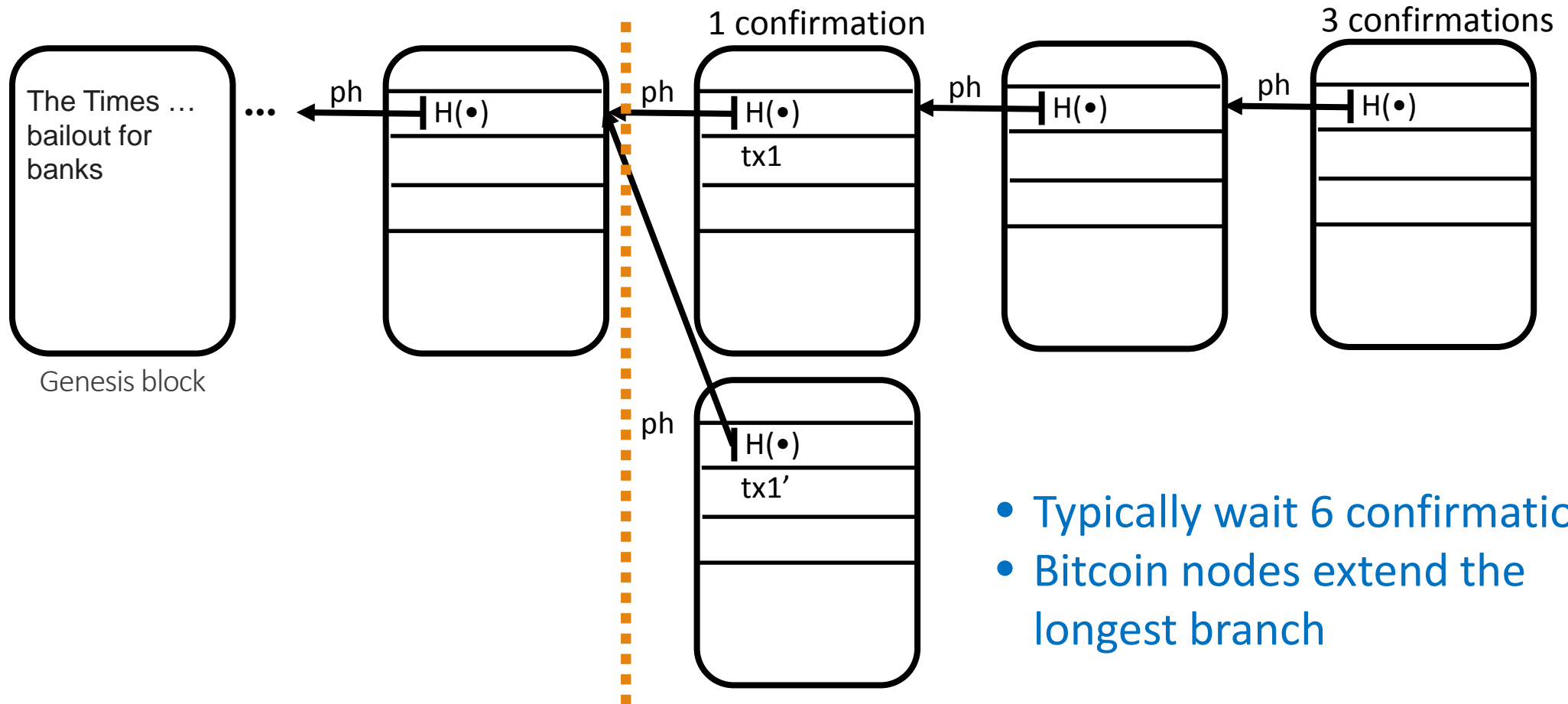


Suppose

- $tx1$  contains a transfer of a particular coin from A to B, and
- $tx1'$  contains a transfer of the same coin from A to C

Which one is “legitimate”?

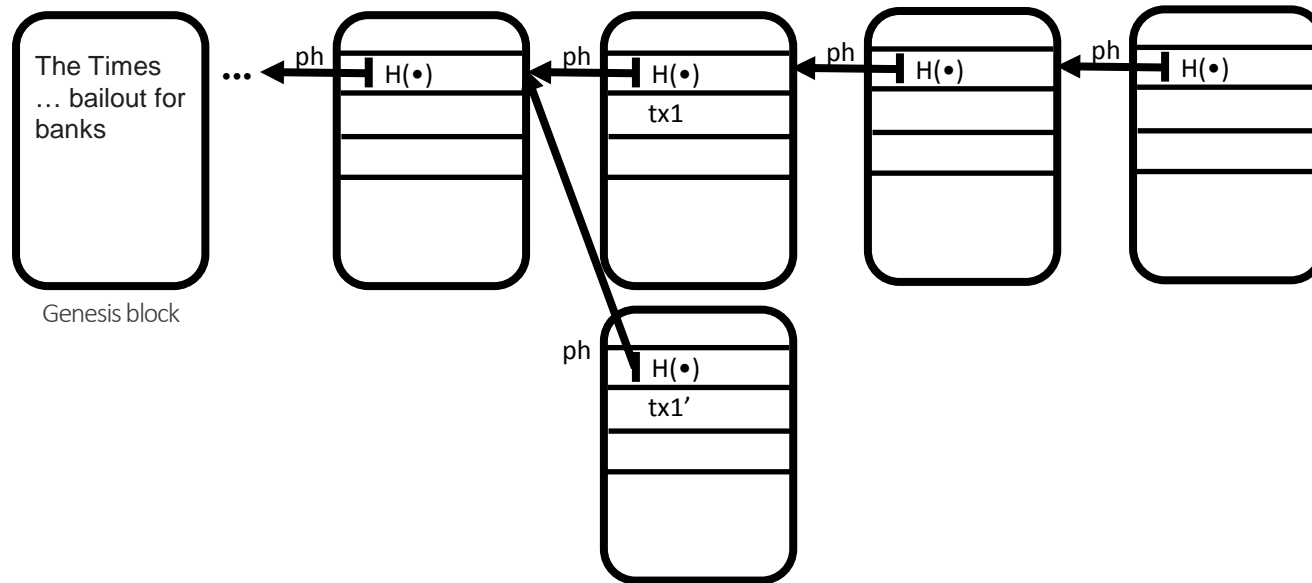
# Double-spending?



- Typically wait 6 confirmations
- Bitcoin nodes extend the longest branch



# Double spending?



- Prevented purely by **consensus**.
- Cryptography has nothing to do with it.

# Discussions

---

- We have discussed
  - The bitcoin network
  - The bitcoin consensus protocol:
    - what the consensus is on
    - how it works
- Next time: Blockchain systems as Byzantine Fault-Tolerant (BFT) distributed systems!

