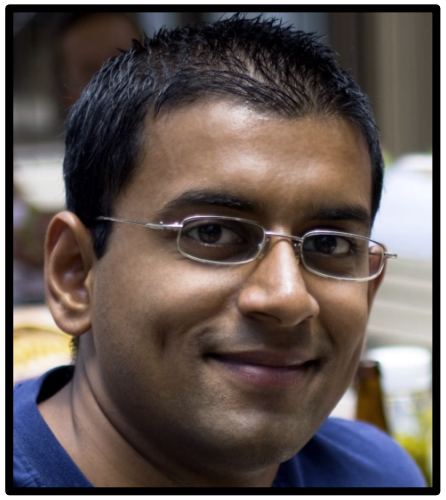
Acknowledgements

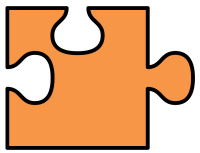
Joseph Bonneau

Ed Felten



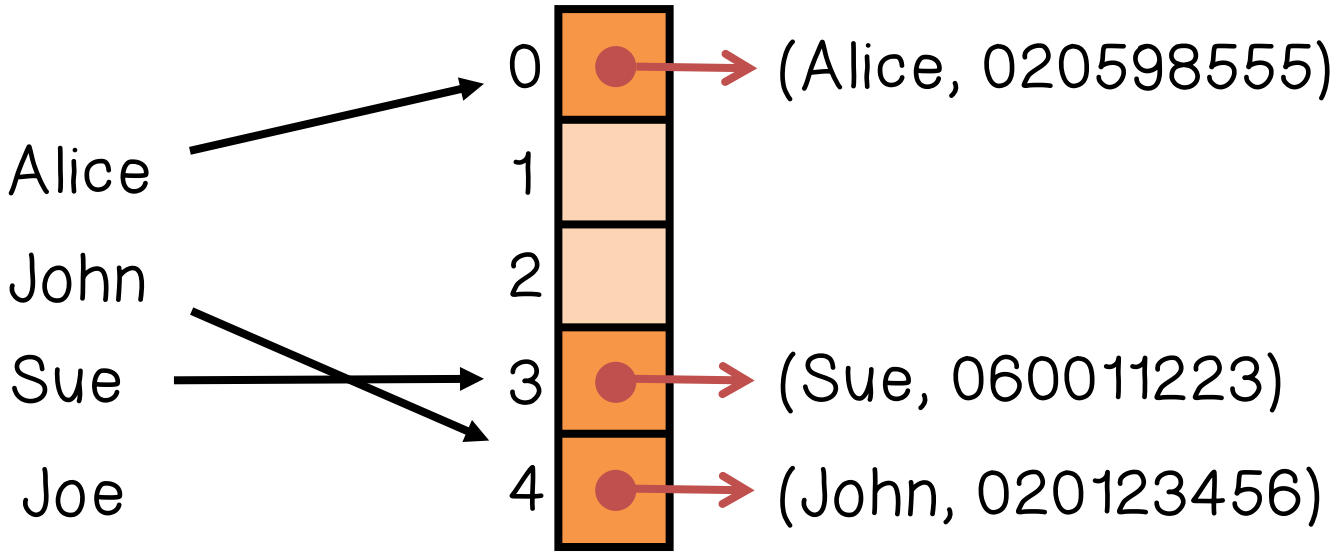
Arvind Narayanan

Andrew Miller



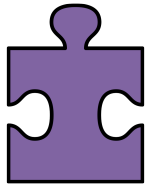
Where Does Joe Go Quiz

h(w) = (length of word w) mod 5



Where does Joe go in the table? 3

What is the weakness of this hash function? Collisions



Hash Function Quiz

With regards to hash functions select all the true statements:

- [x] Hash functions do not have a key
- [x] Hash functions are also called one-way encryption
- [ ] A major drawback of hash functions is the possibility of two messages having the same hash value
- [x] Hash functions are primarily used for message integrity

Review of Hash Functions

Easy to compute H(m)

Compute message digest of data of any size

Fixed length output: 128-512 bits

Review of Hash Functions

Given H(m), no easy way to find m

- One-way function

Given m_1, it is computationally infeasible to find $m_2 \neq m_1$ s.t. $H(m_2) = H(m_1)$

- Weak collision resistant

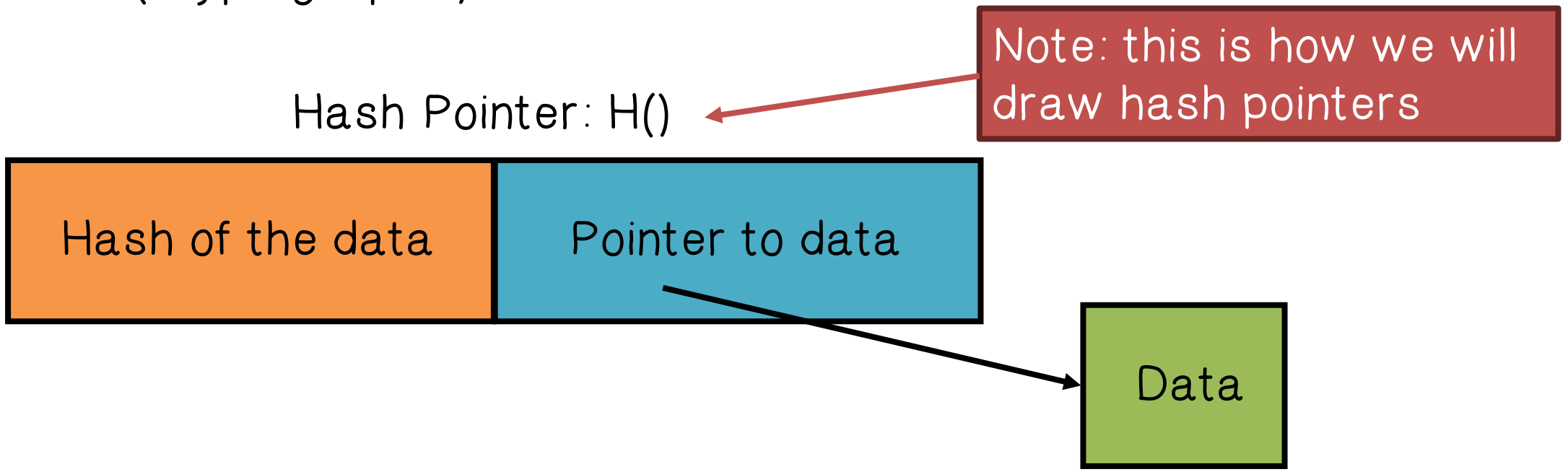
Computationally infeasible to find $m_1 \neq m_2$ s.t. $H(m_1) = H(m_2)$

- Strong collision resistant

Hash Pointers and Data Structures

Hash pointer contains:

- pointer to where some info is stored
- (cryptographic) hash of the info



Hash Pointers and Data Structures

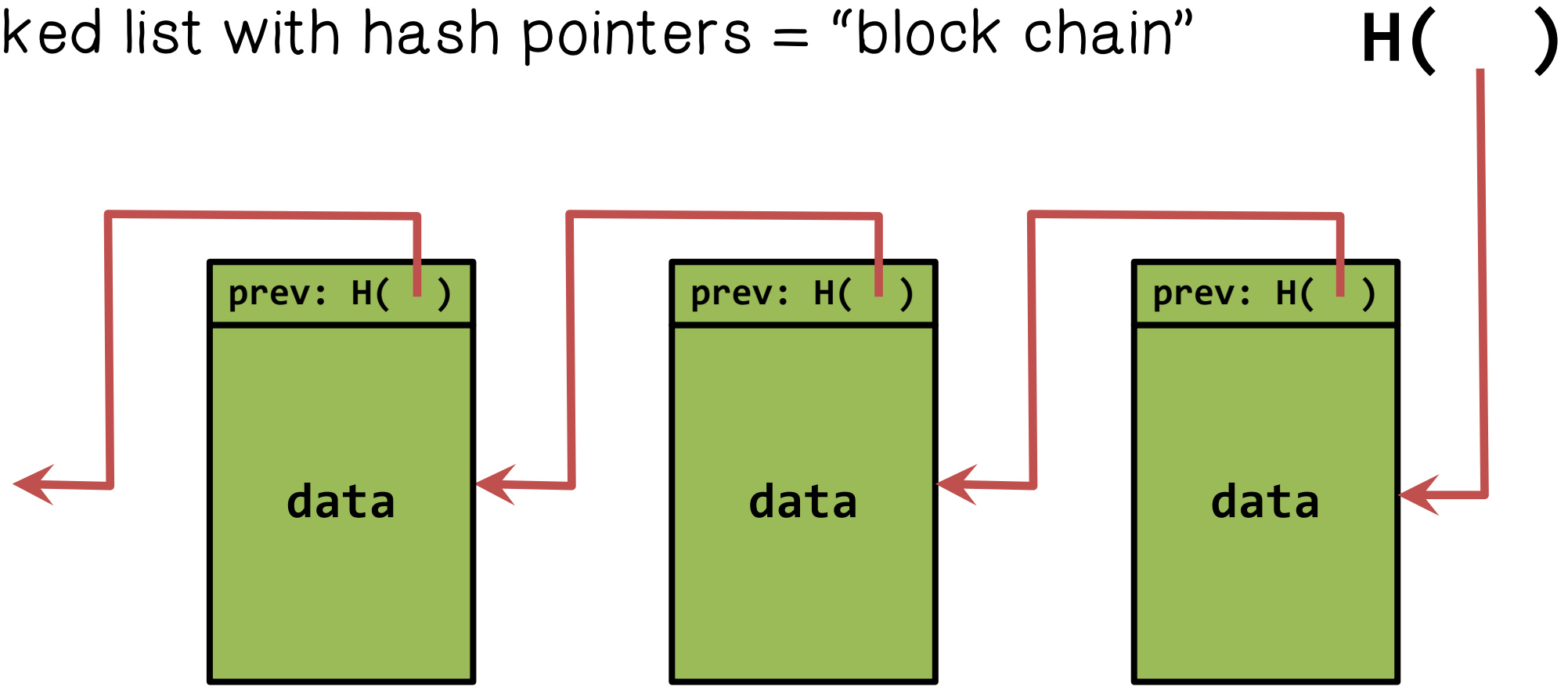
If we have a hash pointer, we can:

- ask to get the info back
- verify that it hasn't changed

Key Idea: build data structures with hash pointers

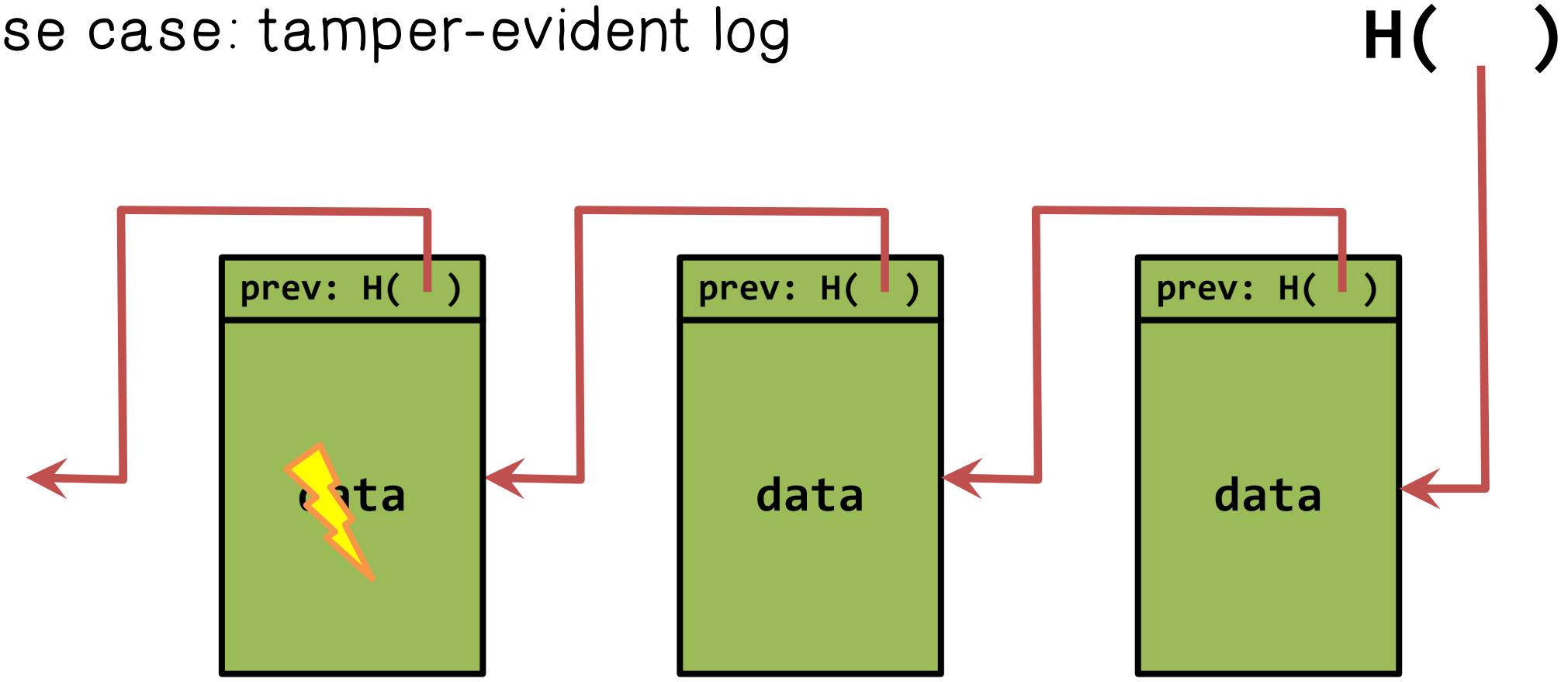
Hash Pointers and Data Structures

linked list with hash pointers = "block chain"

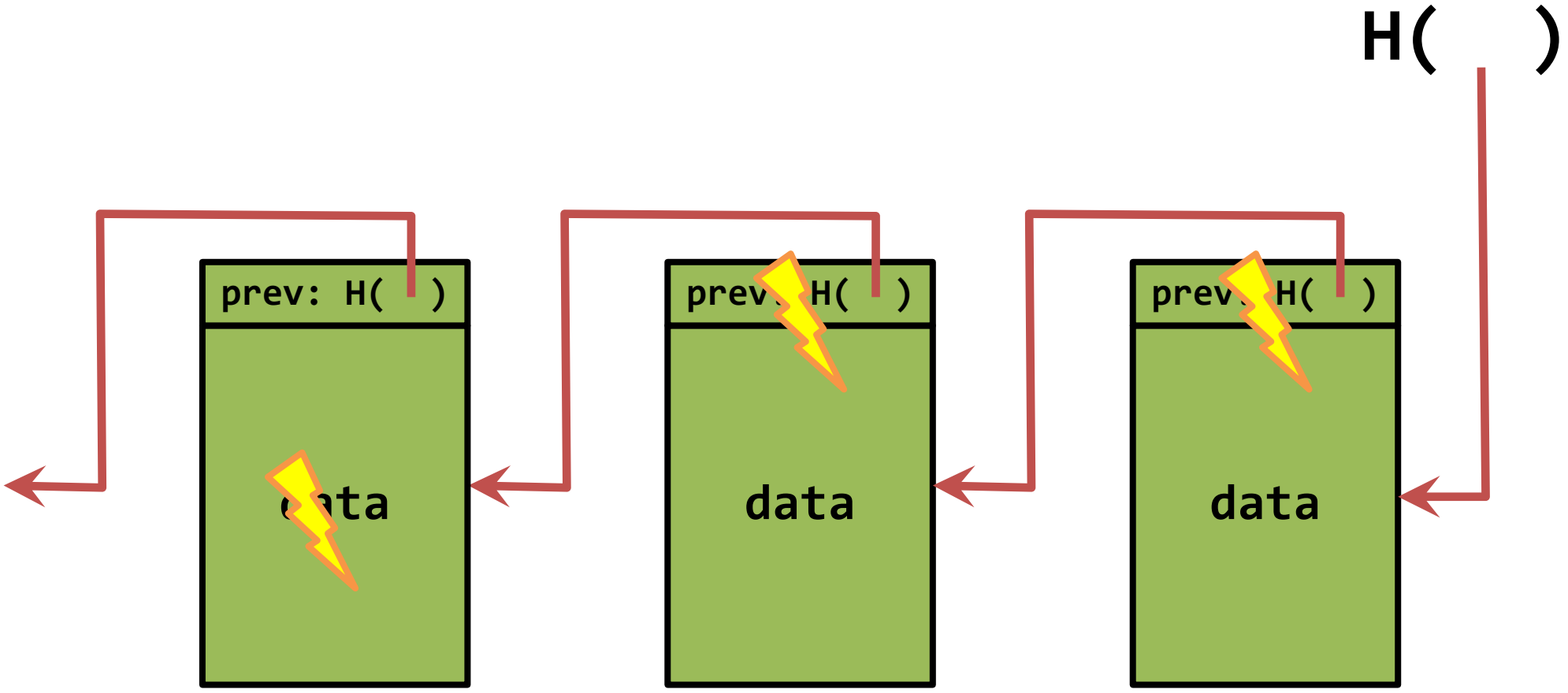


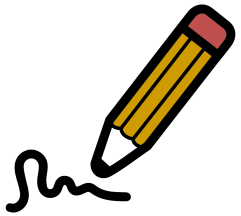
Hash Pointers and Data Structures

Use case: tamper-evident log



Hash Pointers and Data Structures



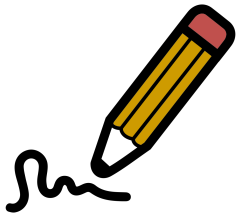


Digital Signatures

What we want from signatures

Only you can sign, but anyone can verify

Signature is tied to a particular document and can't be cut-and-pasted to another document

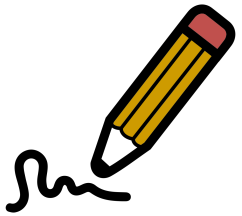


Digital Signatures

API for digital signatures

- sk: secret signing key
- pk: public verification key

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



Digital Signatures

Requirements for signatures

"valid signatures verify"

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

"can't forge signatures"

- adversary who:
  - knows pk
  - gets to see signatures on messages of his choice
  - can't produce a verifiable signature on another message

Public Keys as Identities

Useful trick: public key == an identity

if you see sig such that verify(pk, msg, sig)==true, think of it as

pk says, “[msg]”

to “speak for” pk, you must know matching secret key sk

Public Keys as Identities

How to make a new identity

create a new, random key-pair (sk, pk)

- pk is the public "name" you can use **[**usually better to use Hash(pk)**]**
- sk lets you "speak for" the identity

you control the identity, because only you know sk if pk "looks random", nobody needs to know who you are

Public Keys as Identities

Decentralized Identity Management

Anybody can make a new identity at any time

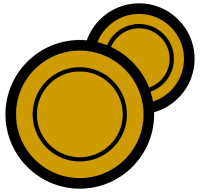
- make as many as you want!
- no central point of coordination

These identities are called “addresses” in Bitcoin.

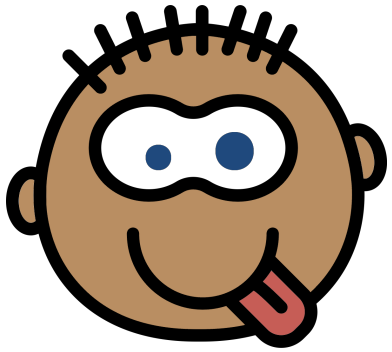
Cryptocurrency Quiz

Select all the true statements with regards to cryptocurrency:

- [x] The security of crypto currency ledgers depends on the honesty of its miners.
- [ ] Most cryptocurrencies are designed to maintain production, to keep inflation in check
- [ ] Since cryptocurrencies are pseudo-anonymous it is okay that they are more susceptible to law enforcement seizure.

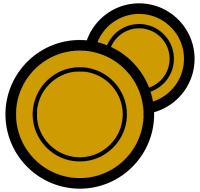


Simple Cryptocurrencies

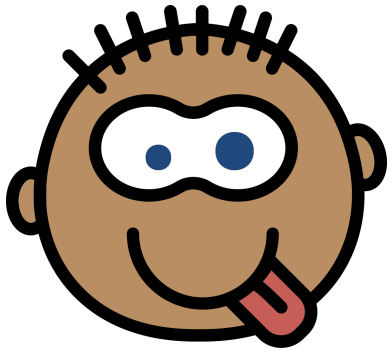


GoofyCoin

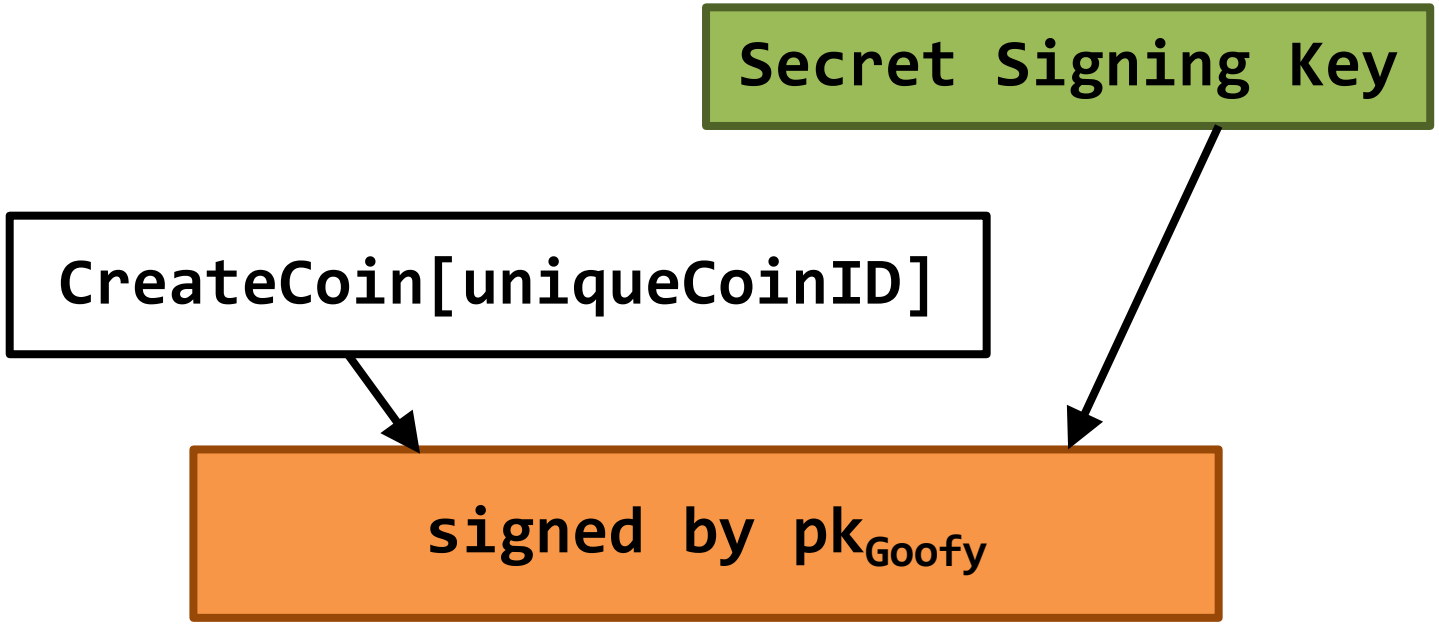
- Goofy can create new coins
- New coins belong to the creator

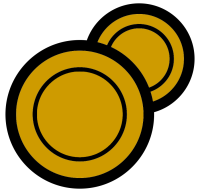


Simple Cryptocurrencies

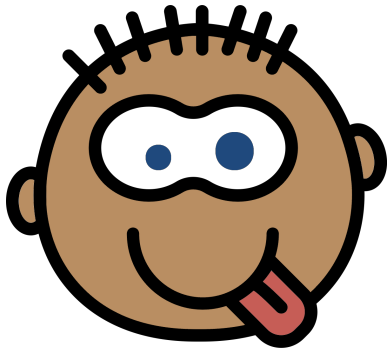


GoofyCoin

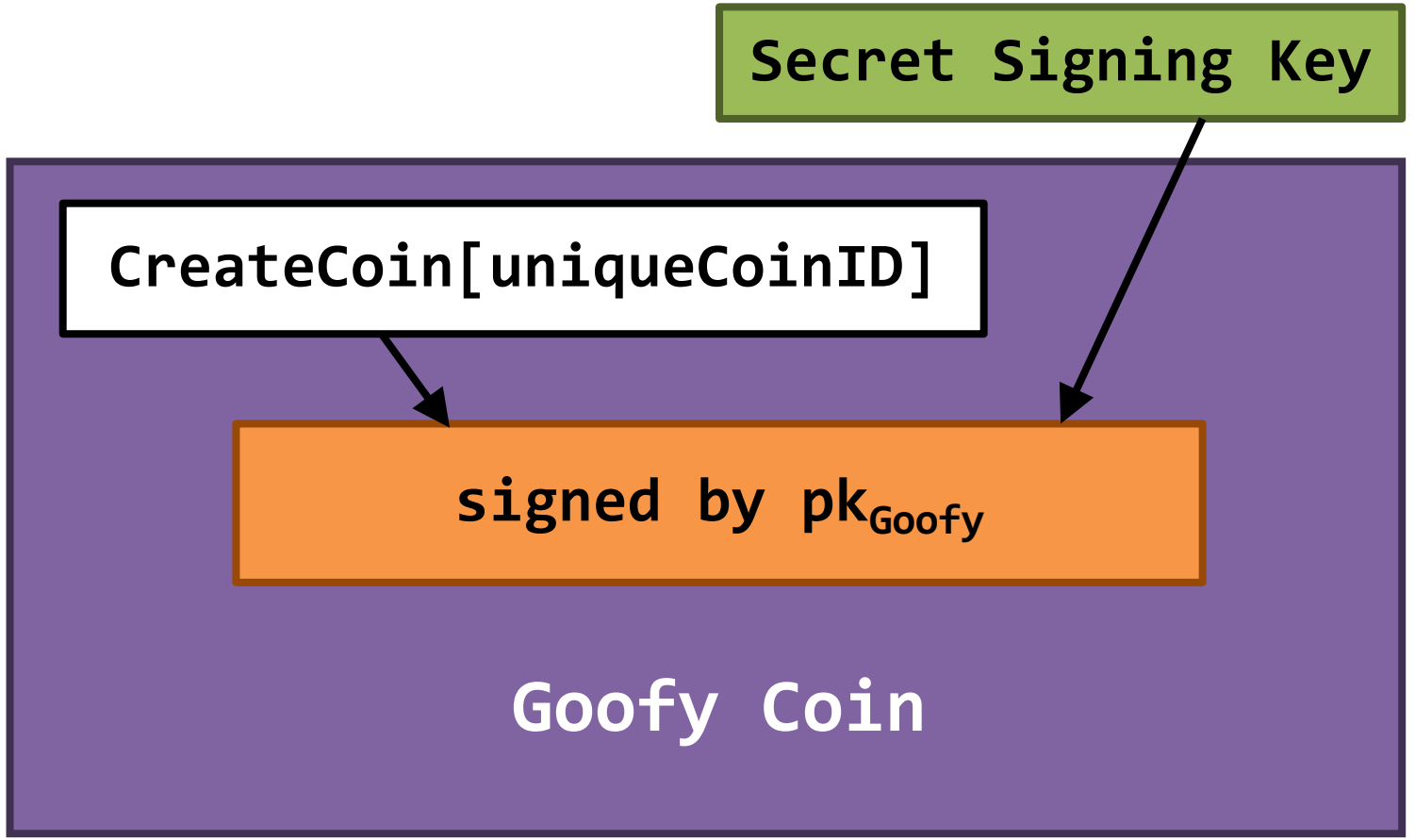


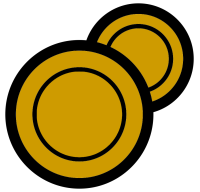


Simple Cryptocurrencies

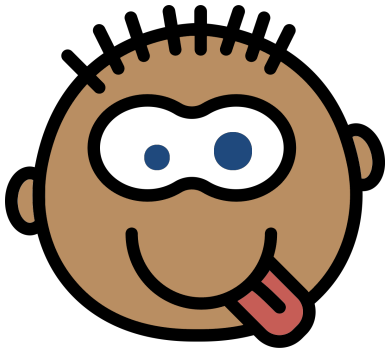


GoofyCoin



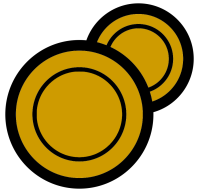


Cryptocurrency

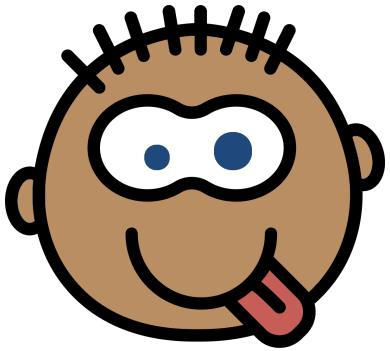


Goofy

A coin's owner can spend it - using cryptographic operations



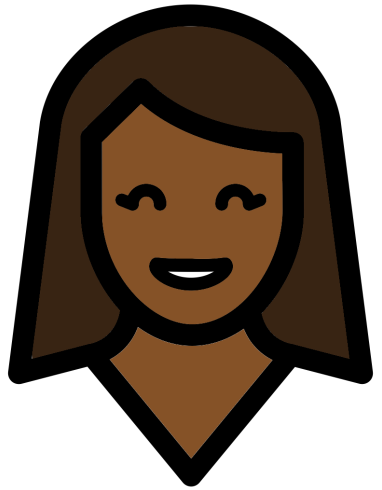
Cryptocurrency



Goofy

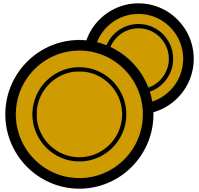
signed by pk_{Goofy}
Pay to pk_{Alice} : H( )

signed by pk_{Goofy}
CreateCoin [uniqueCoinID]



Alice

Alice owns it now.



Cryptocurrency

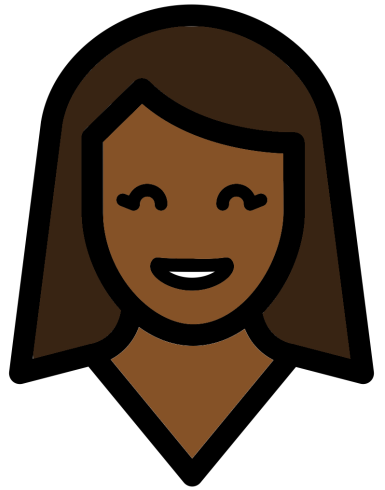
The recipient can pass on the coin again.

signed by pk_{Alice}
Pay to pk_{Bob} : H( )

signed by pk_{Goofy}
Pay to pk_{Alice} : H( )

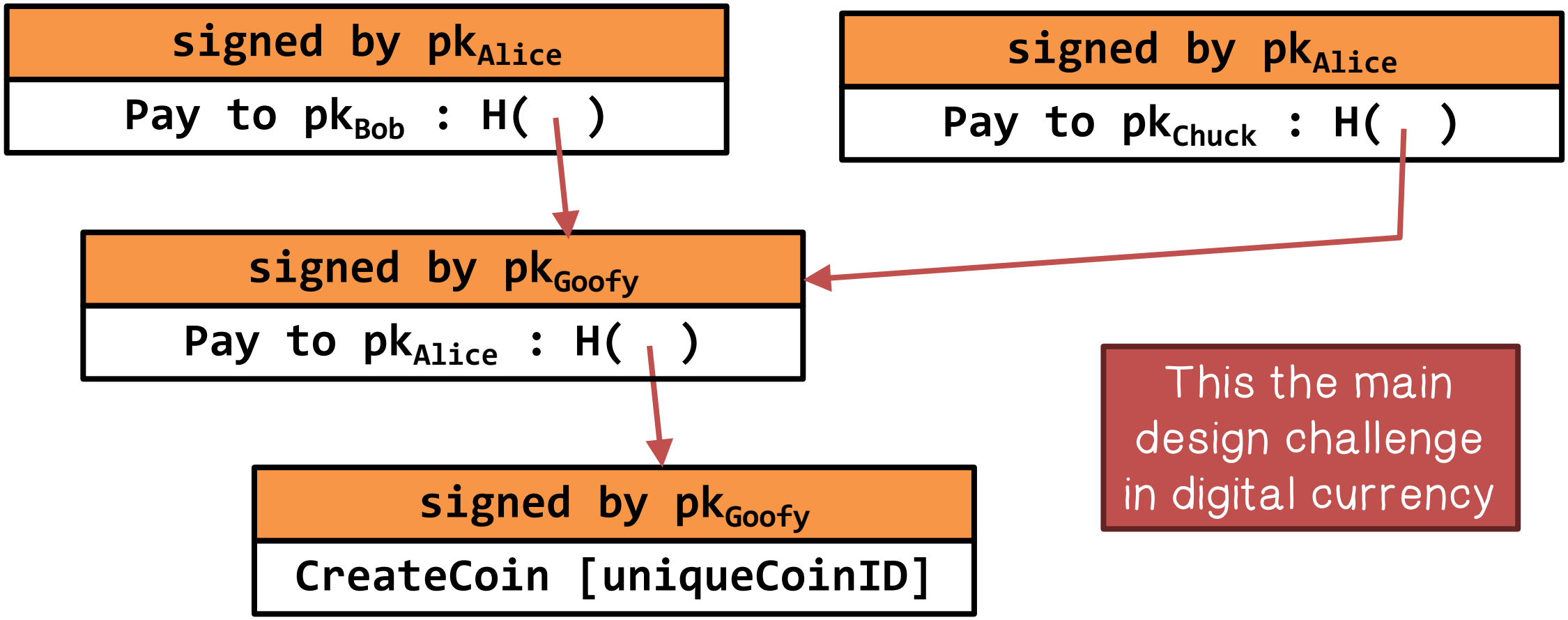
signed by pk_{Goofy}
CreateCoin [uniqueCoinID]

Bob owns it now.



Alice

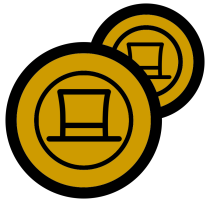
Double Spending Attack



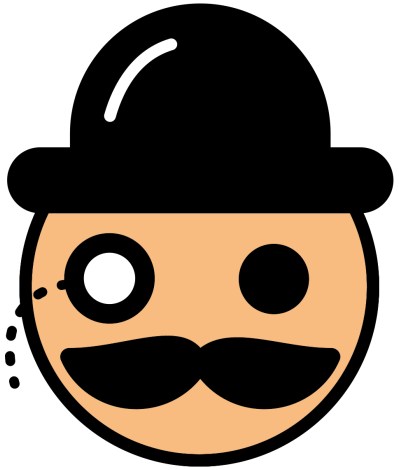
Wallet Quiz

Match the characteristic to each wallet.
Answer choices are: Cold, Desktop, Hardware, Hot, Mobile, Online

Hot wallet	a wallet connected to the internet
Cold wallet	the wallet is offline
Desktop wallet	used on laptops and personal computers
Mobile wallets	QR code capable, with instant payments
Online wallet	wallets provided on the cloud
Hardware wallet	developers make use of top grade cryptography



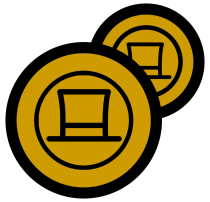
Scrooge Coin



Scrooge

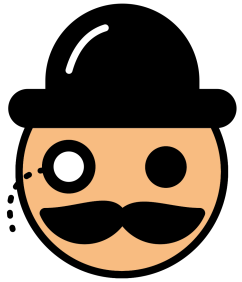
A designated entity publishes an append-only ledger containing the history of all the transactions that have happened

All transactions be written to the ledger before they are accepted



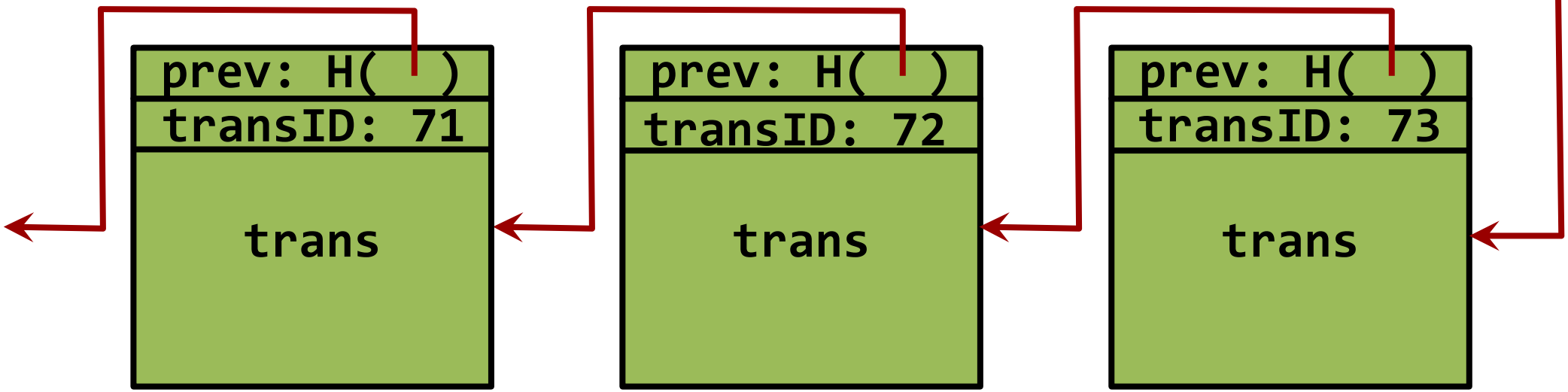
Scrooge Coin

Scrooge publishes a history of all transactions
(a block chain, signed by Scrooge)

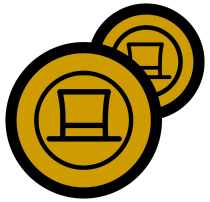


Scrooge

H( )



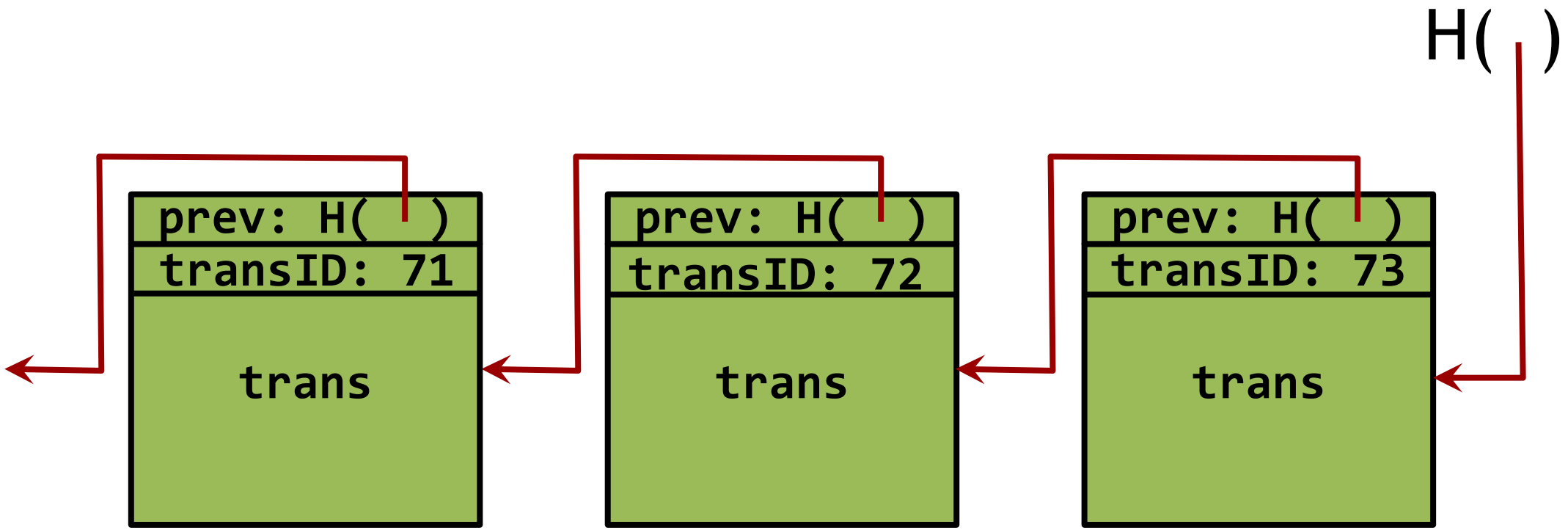
optimization: put multiple transactions in the same block

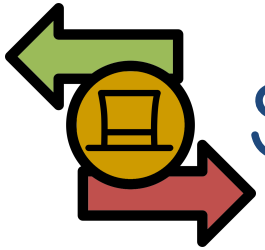


Scrooge Coin

Block chain with hash pointers plus signature = append only property

Scrooge Coin





Scrooge Transactions

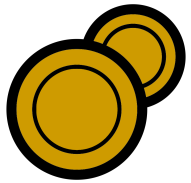
CreateCoins transaction creates new coins

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

coinID 73(0)

coinID 73(1)

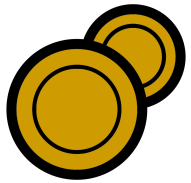
coinID 73(2)



PayCoins

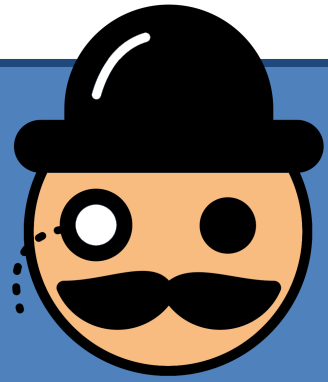
PayCoins transaction consumes (and destroys) some coins, and creates new coins of the same total value

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



PayCoins

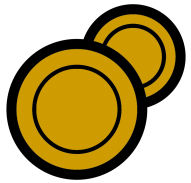
PayCoins transaction consumes (and destroys) some coins, and creates new coins of the same total value



Scrooge

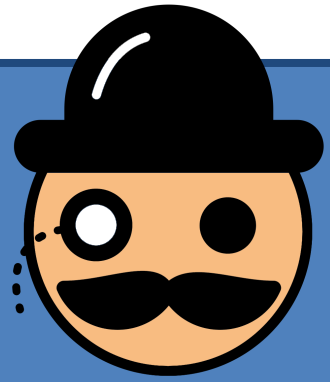
Valid if:

- consumed coins valid,
- not already consumed,
- total value out = total value in, and
- signed by owners of all consumed coins



PayCoins

PayCoins transaction consumes (and destroys) some coins, and creates new coins of the same total value



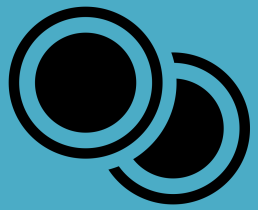
Scrooge

Valid if:

- consumed coins valid,
- not already consumed,
- total value out = total value in, and
- signed by owners of all consumed coins

Immutable Coins

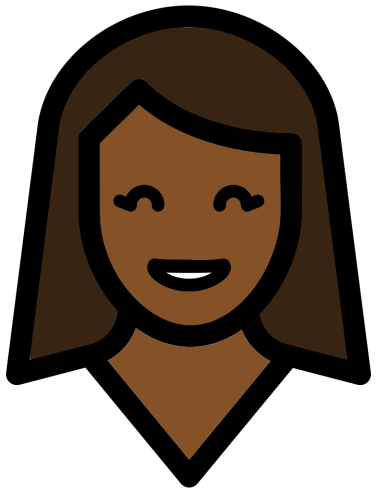
Can never be changed, subdivided, or combined



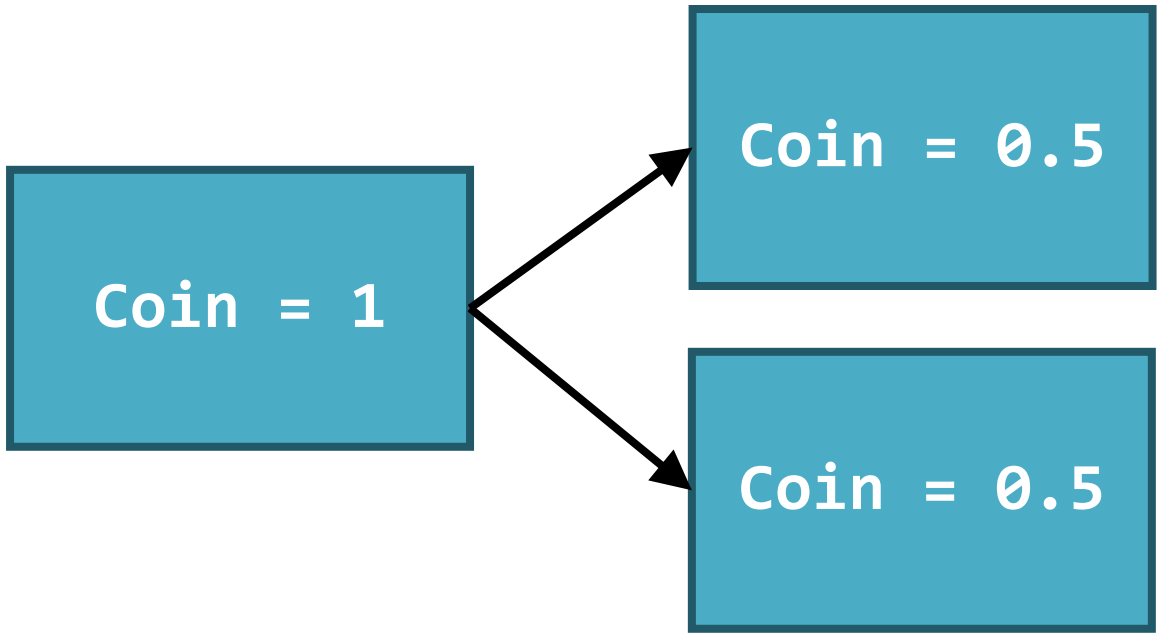
Each coin is created and consumed

Use transactions to subdivide or combine

Immutable Coins

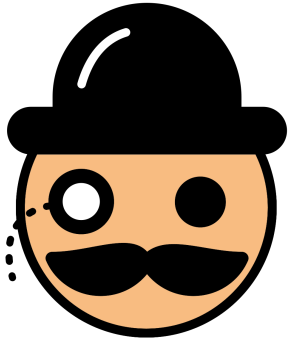


Alice



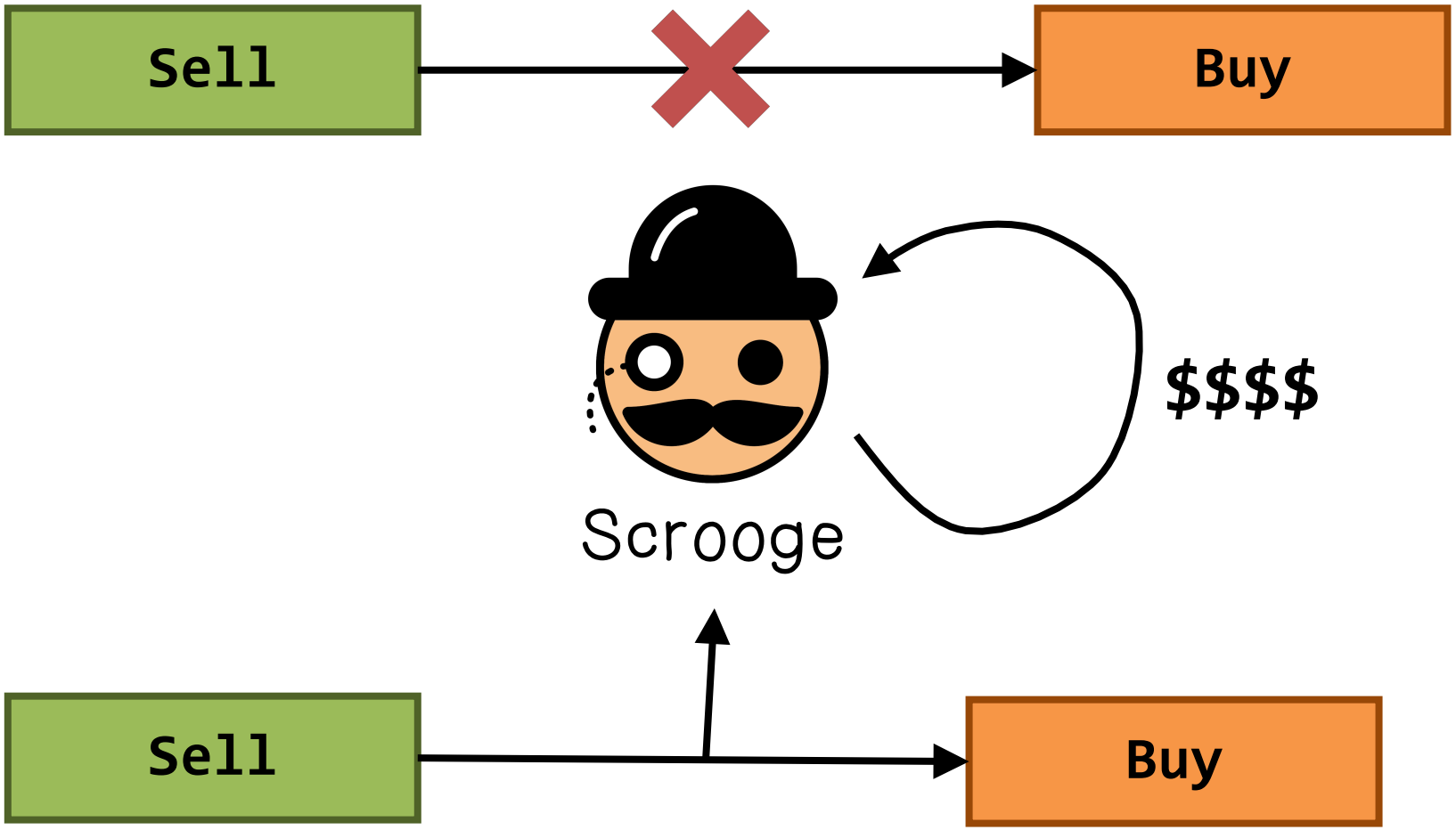
Centralization Problem

Prevents double spending



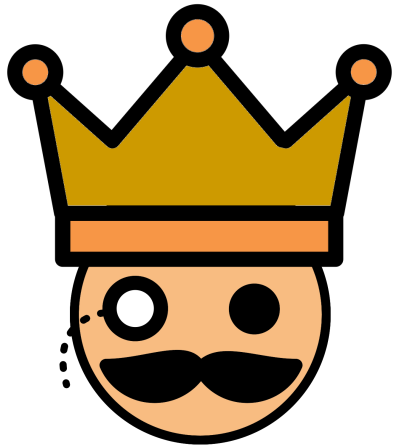
Scrooge

Centralization Problem



Centralization Problem

Cryptocurrencies with central authority have failed to take off



Scrooge

Can we de-centralize a cryptocurrency?

Low acceptance of a cryptocurrency with a centralized authority

Centralization Problem

To Decentralize:

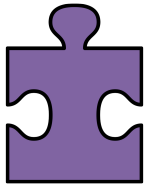
- A single published blockchain with a history of transactions
- Agreement on which transactions are valid
- Agreement on which transactions have occurred
- Decentralized ID assignment
- Decentralized mining of new coins

Bitcoin

Top Cryptocurrency Failures Quiz

Match the cryptocurrency description to its name.
The answer choices are: GetGems. Paycoin, DAO (decentralized Autonomous Org), Dogecoin, Spacebit

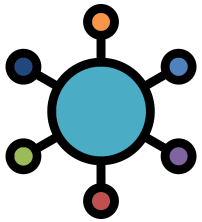
Spacebit	provide a globally accessible block chain through the use of non-satellites.
GetGems	a social networking platform that uses cryptocurrency to members that view ads in the app. Popular in Uzbekistan.
Dogecoin	a decentralized peer-to-peer digital current. The community is friendly and vibrant and known for charitable acts, such as sending the 2014 Jamaican bobsled team to the Olympics.
Paycoin	according to a published white paper, it used new variations on the block chain that would result in a new breed of cryptocurrency.
DAO	the largest crowdfund in history. An attacker exploited a vulnerability in its smart contract, with losses totaling $50 million.



Top Cryptocurrency Failures Quiz

- There is a strong market for it
- It is a volatile field
- It is not an easy problem to solve

Will cryptocurrency become obsolete with banks using blockchain?



Bitcoins and Decentralization

Who maintains the ledger?

Who has authority over which transactions are valid?

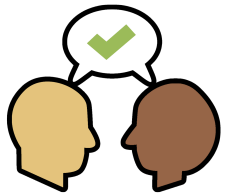
Who creates new bitcoins?

Who determines how the rules of the system change?

How do bitcoins acquire exchange value?

Beyond the protocol:

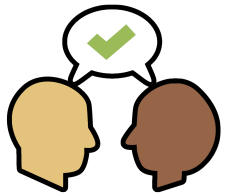
- exchanges, wallet software, service providers…



Distributed Consensus

The protocol terminates and all correct nodes decide on the same value

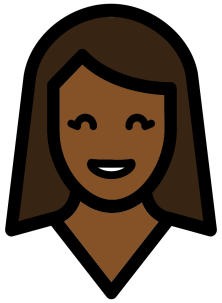
This value must have been proposed by some correct node



Distributed Consensus

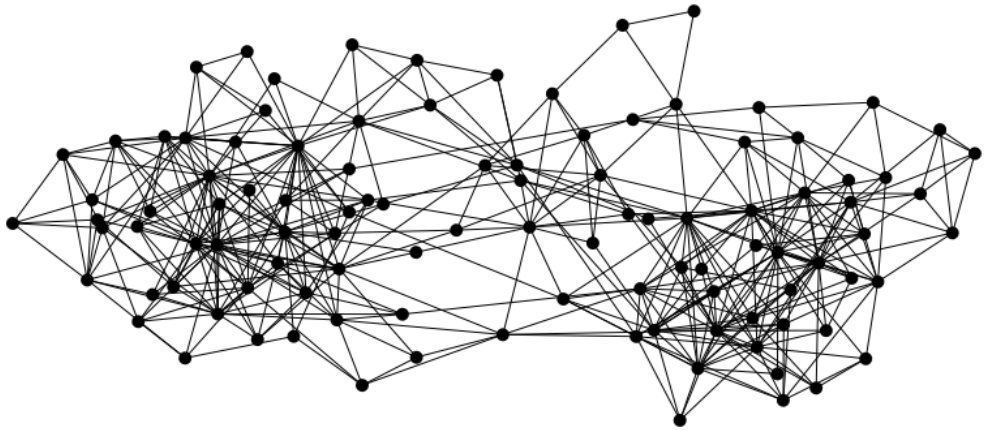
Bitcoin is a peer-to-peer system

When Alice wants to pay Bob:
she broadcasts the transaction to all Bitcoin nodes

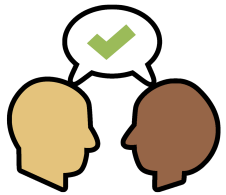


Alice

signed by Alice
Pay to pk_{Bob} : H( )



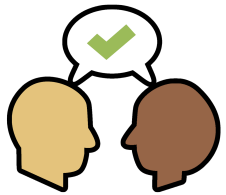
(Note: Bob's computer is not in the picture)



Distributed Consensus

Why consensus is hard:

- Nodes may crash
- Nodes may be malicious
- Network is imperfect
  - Not all pairs of nodes connected
  - Faults in network
  - Latency
    - No notion of global time



Distributed Consensus

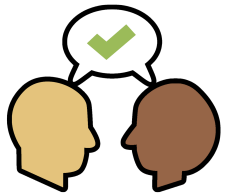
Bitcoin Consensus

Introduces incentives

- Possible only because it's a currency!

Embraces randomness

- Does away with the notion of a specific end-point
- Consensus happens over long time scales — about 1 hour



Distributed Consensus

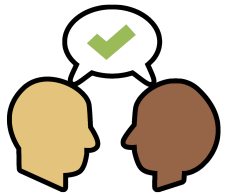
In each round, random node is picked

This node proposes the next block in the chain

Other nodes implicitly accept/reject this block

- by either extending it
- or ignoring it and extending chain from earlier block

Every block contains hash of the block it extends



Distributed Consensus

Consensus Algorithm(simplified)

New transactions are broadcast to all nodes

Each node collects new transactions into a block

In each round a random node gets to broadcast its block

Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)

Nodes express their acceptance of the block by including its hash in the next block they create

Bitcoin Safeguards

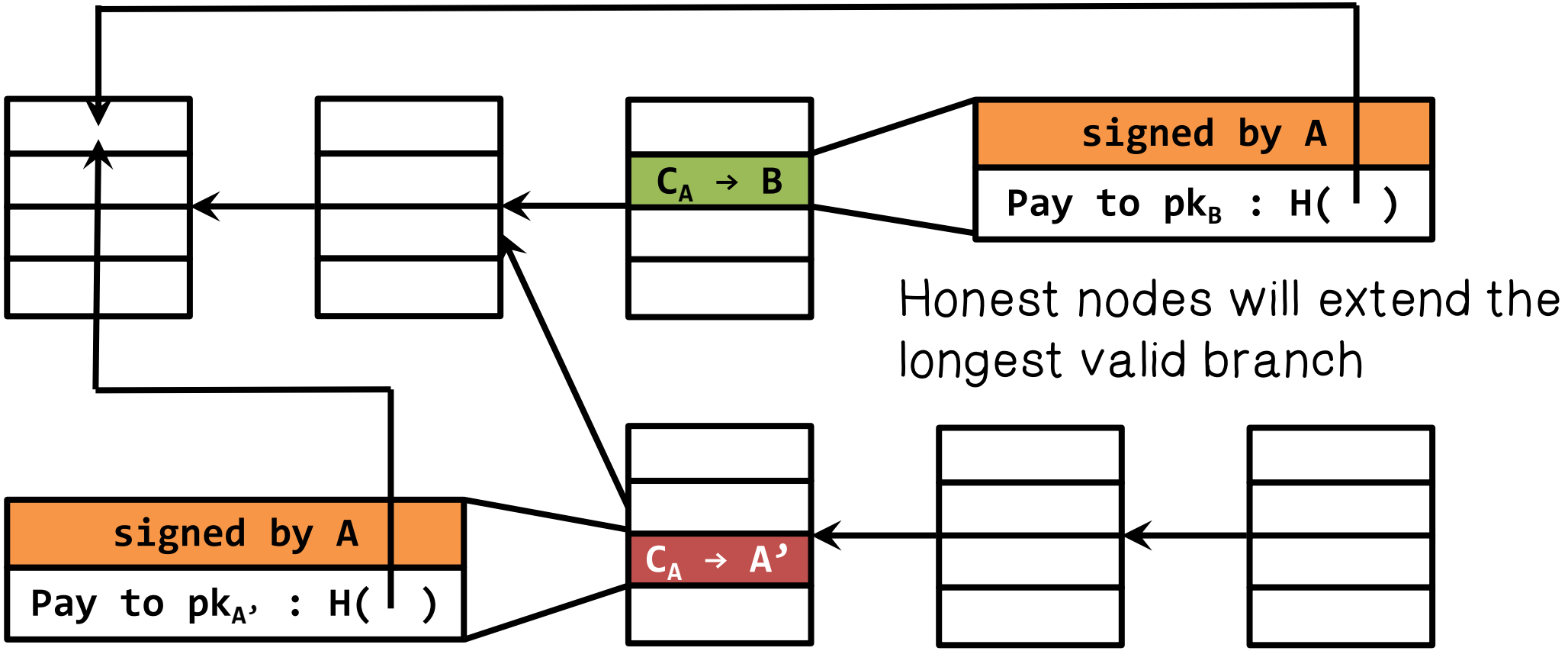
What can a malicious node do?

Stealing a user's bitcoin requires the attacker create a valid transaction that would spend the coin.

A valid transaction requires a digital signature, which cannot be forged if the underlying cryptography is solid.

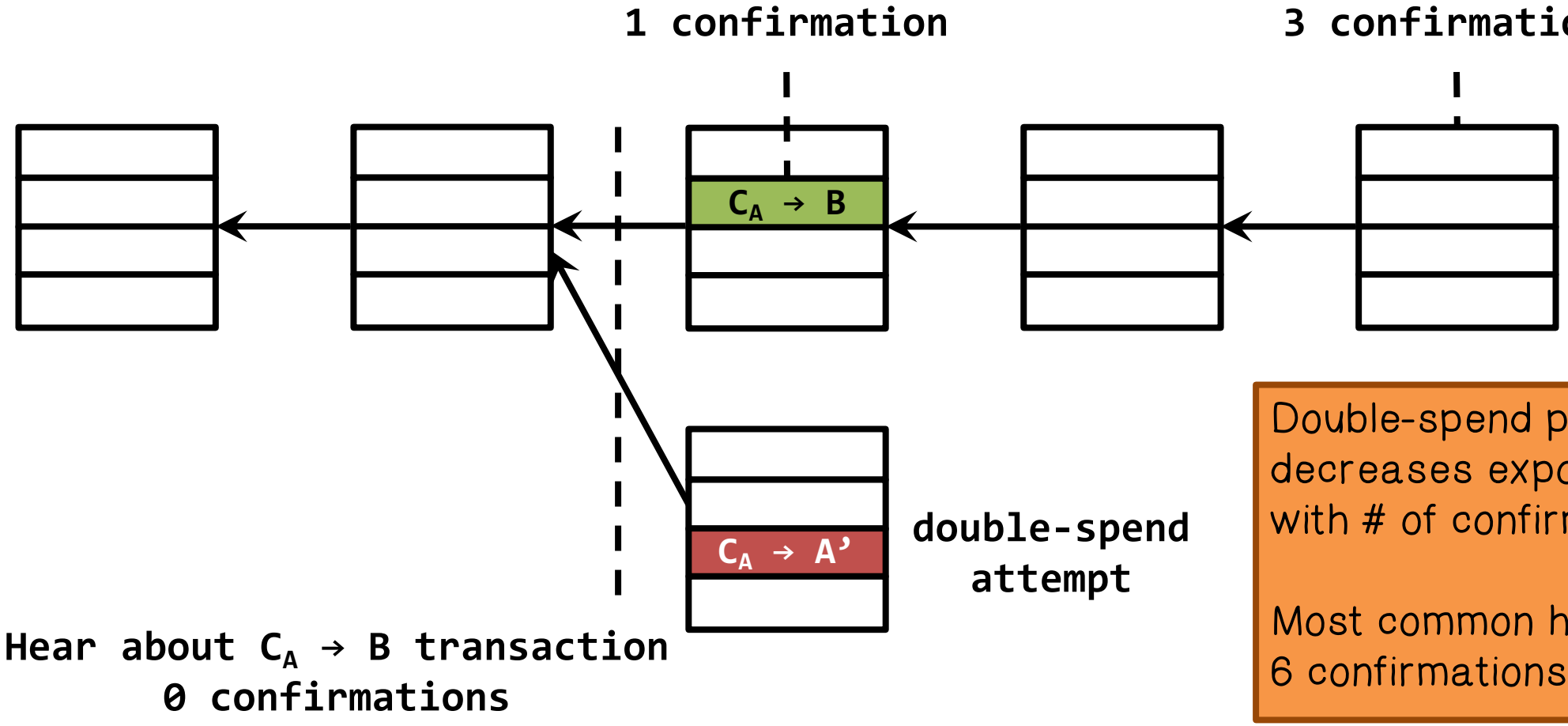
Bitcoin Safeguards

Double Spending Attack



Bitcoin Safeguards

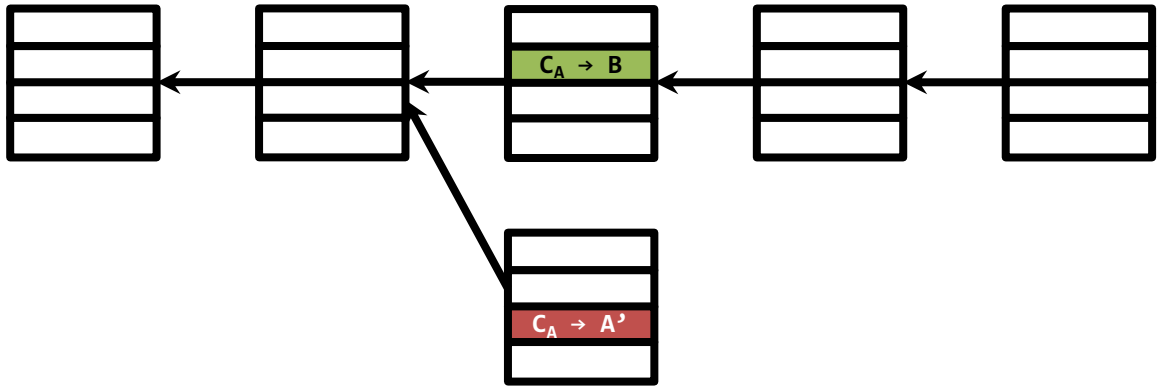
Double Spending Attack from the merchant's point of view



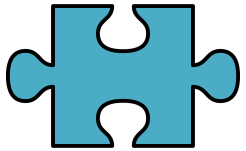
Double-spend probability decreases exponentially with # of confirmations

Most common heuristic: 6 confirmations

Bitcoin Safeguards



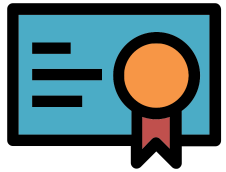
- Protection against invalid transactions is cryptographic, but enforced by consensus
- Protection against double-spending is purely by consensus
- You're never 100% sure a transaction is in consensus branch. Guarantee is probabilistic



Proof of Work Quiz

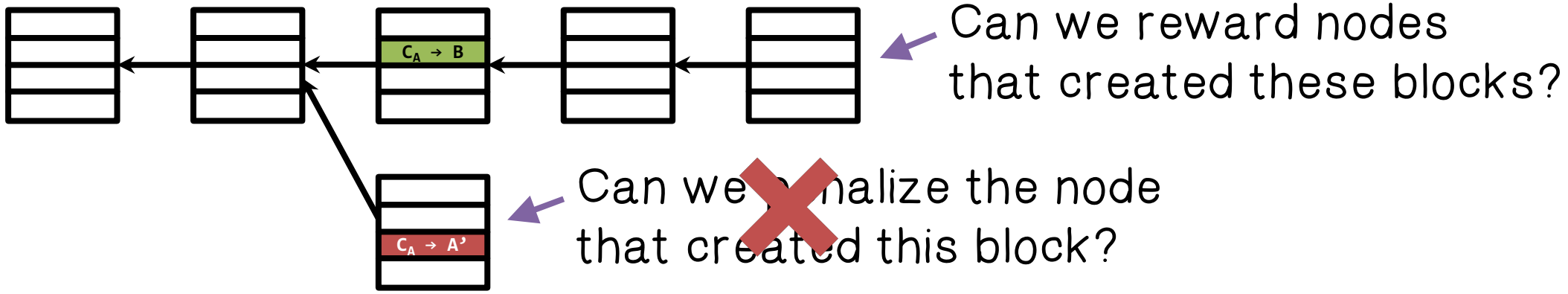
With regards to Bitcoin, which of the following statements are true?

- [x] Proof of work is costly and time consuming to produce
- [ ] Proof of work is costly and time consuming to verify
- [ ] To earn a coin, miners of bitcoins must complete some the work in the block
- [x] Changing a block requires regenerating all successors and redoing the work they contain.



Incentives and Proof of Work

- Assumption of honesty is problematic
- Can we give nodes incentives for behaving honestly?



- Everything so far is just a distributed consensus protocol
- But now we utilize the fact that the currency has value

Bitcoin Incentive #1

Creator of block gets to

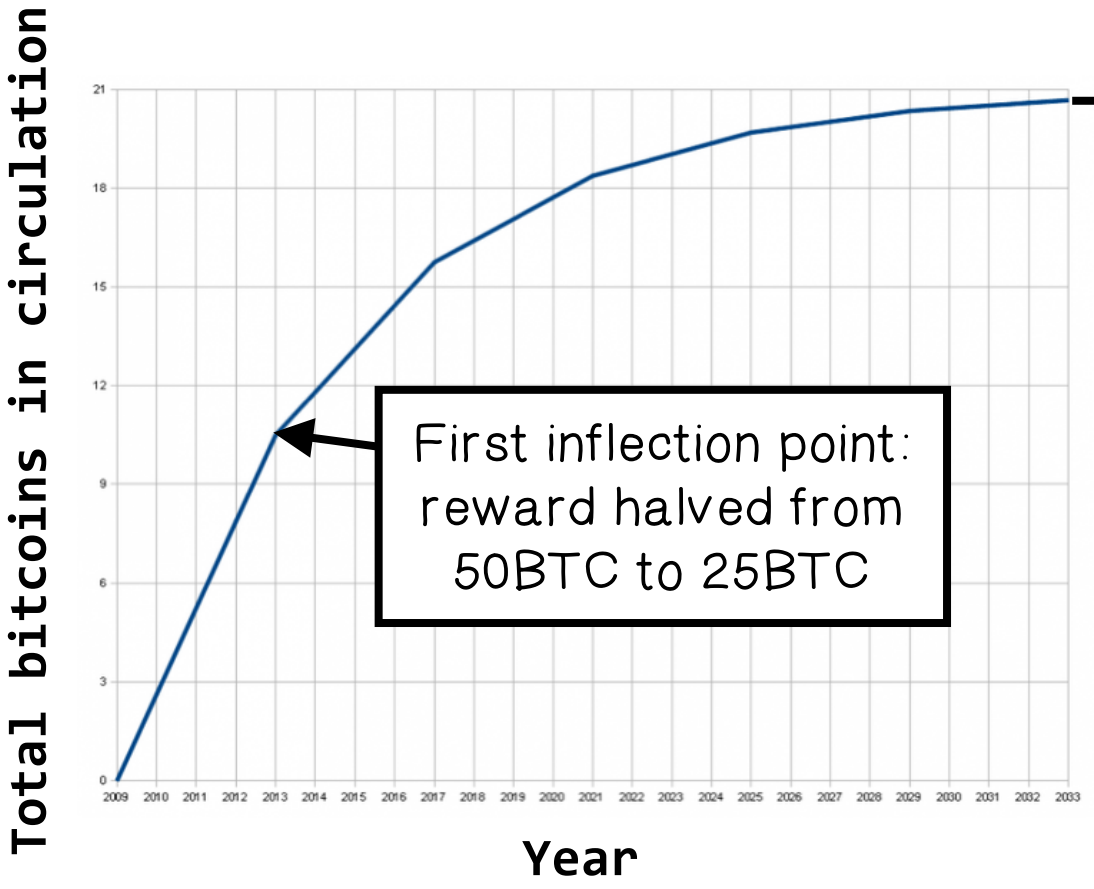
- include special coin-creation transaction in the block
- choose recipient address of this transaction

Value is fixed: currently 25 BTC, halves every 4 years

Block creator gets to “collect” the reward only if the block ends up on long-term consensus branch!

Bitcoin Incentive #1

There is a finite supply of bitcoins



Block reward is how new bitcoins are created

Runs out in 2040. No new bitcoins unless rules change

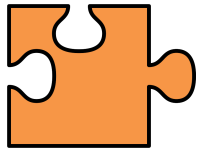
Bitcoin Incentive #2

Transaction Fees

Creator of transaction can choose to make output value less than input value

Remainder is a transaction fee and goes to block creator

Purely voluntary, like a tip



Sybil Attack Quiz

With regards to Sybil attacks, check all true statements:

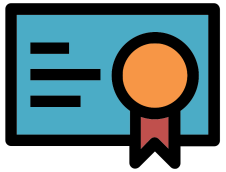
- [x] The attacker creates a lot of fake identities and uses them to change voting outcomes or control the network
- [x] A Sybil attack is designed to attack reputation systems in a peer-to-peer network
- [x] Sybil attack can be stopped if users are willing to give up anonymity.

Bitcoin Remaining Problems

How to pick a random node?

How to avoid a free-for-all due to rewards?

How to prevent Sybil attacks?



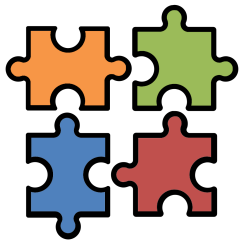
Proof of Work

To approximate selecting a random node:

- select nodes in proportion to a resource that no one can monopolize (we hope)

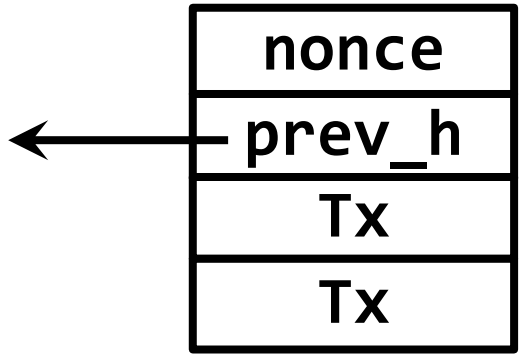
In proportion to computing power: proof-of-work

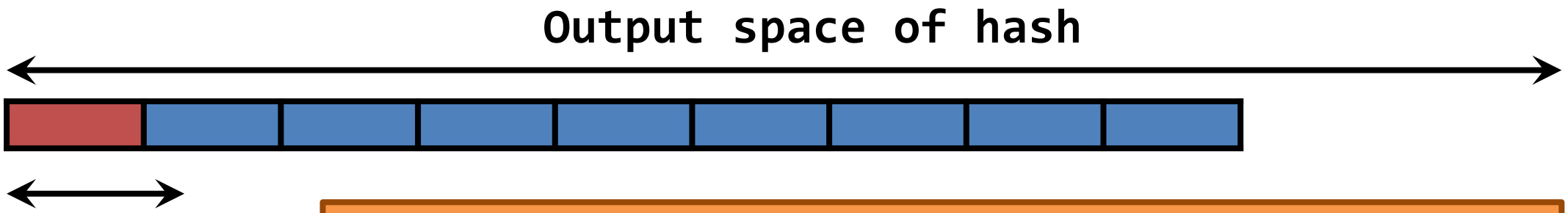
In proportion to ownership: proof-of-stake



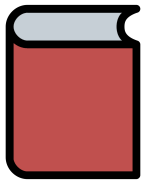
Hash Puzzles

To create block, find nonce s.t.
H(nonce || prev_hash || tx || ... || tx) is very small





If hash function is secure:
Only way to succeed is to try enough nonces until you get lucky

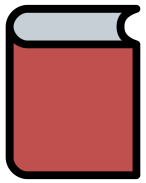


PoW Properties

PoW 1: Difficult to Compute

As of Aug 2014: about 10^{20} hashes/block

Only some nodes bother to compete — miners



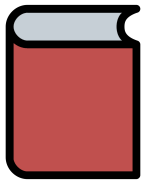
PoW Properties

PoW 2:; Parameterizable Cost

Nodes automatically re-calculate the target every two weeks

Goal: average time between blocks = 10 minutes

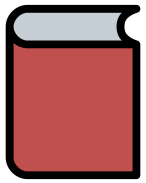
Prob (Alice wins next block) =
fraction of global hash power she controls



PoW Properties

Key Security Assumptions

Attacks infeasible if majority of miners weighted by hash power follow the protocol



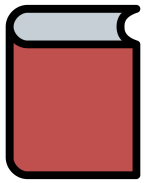
PoW Properties

PoW 3: Trivial to Verify

Nonce must be published as part of block

Other miners simply verify that

$$H(\text{nonce} \,\|\, \text{prev\_hash} \,\|\, \text{tx} \,\|\, \ldots \,\|\, \text{tx}) < \text{target}$$



PoW Properties

Mining Economics

If mining reward (block reward + Tx fees)	>	hardware + electricity cost	→	Profit

Complications:

- fixed vs. variable costs
- reward depends on global hash rate

Bitcoin Summation

What can a "51% attacker" do?

Steal coins from existing address?	No
Suppress some transactions from the block chain?	Yes
Suppress some transactions from the P2P network?	No
Change the block reward?	No
Destroy confidence in Bitcoin?	YES

Bitcoin Summation

Remaining Questions:

How do we get from consensus to currency?

What else can we do with consensus?