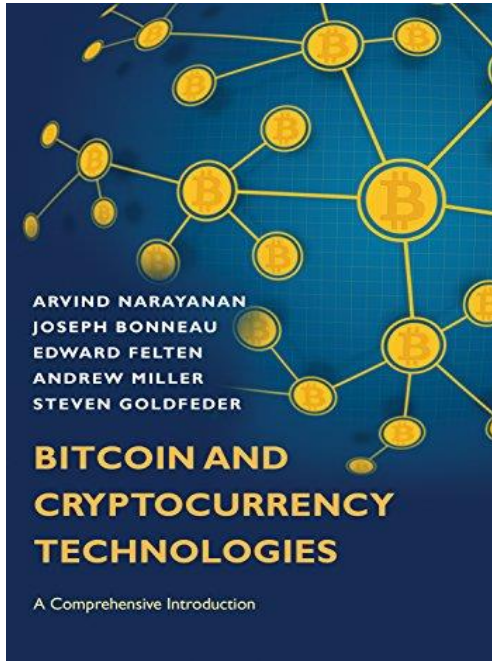


Intro to Crypto and Cryptocurrencies



Slides by Arvind Narayanan et al.

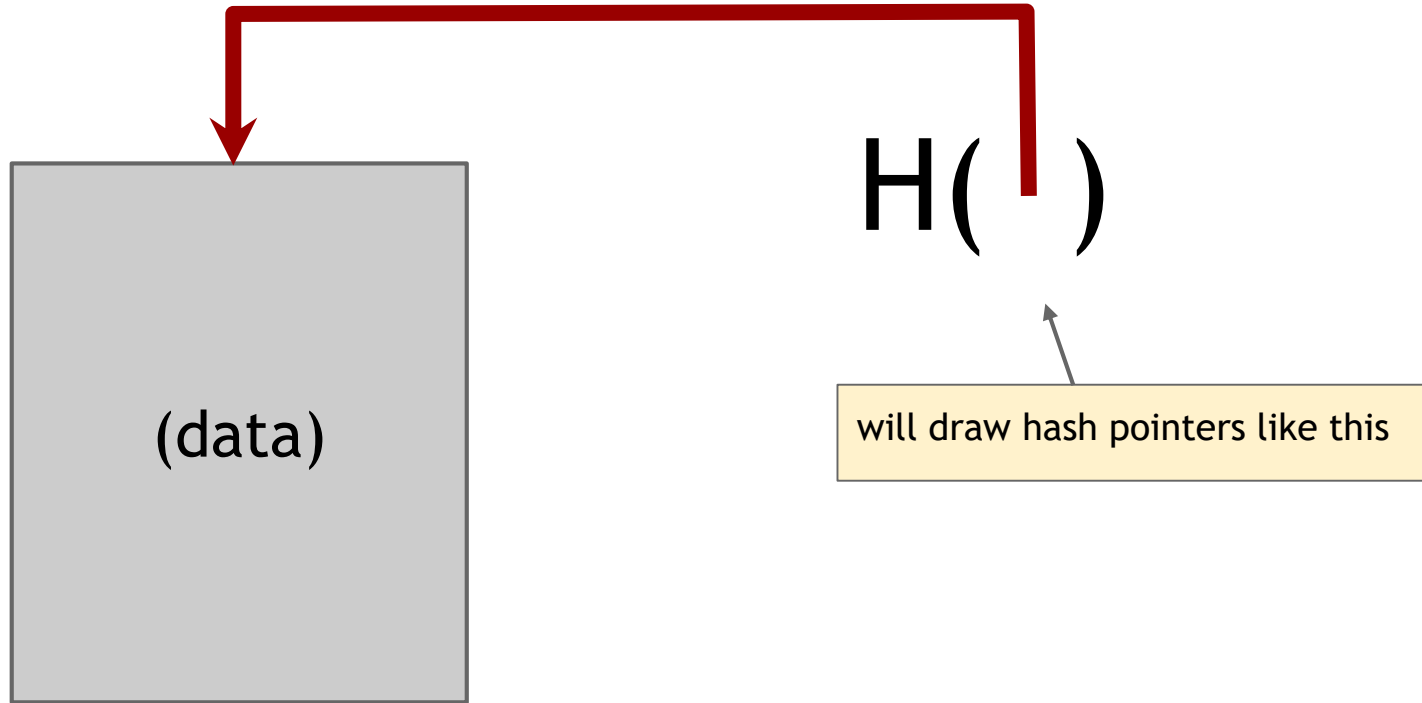
Hash Pointers and Data Structures

hash pointer is:

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

if we have a hash pointer, we can

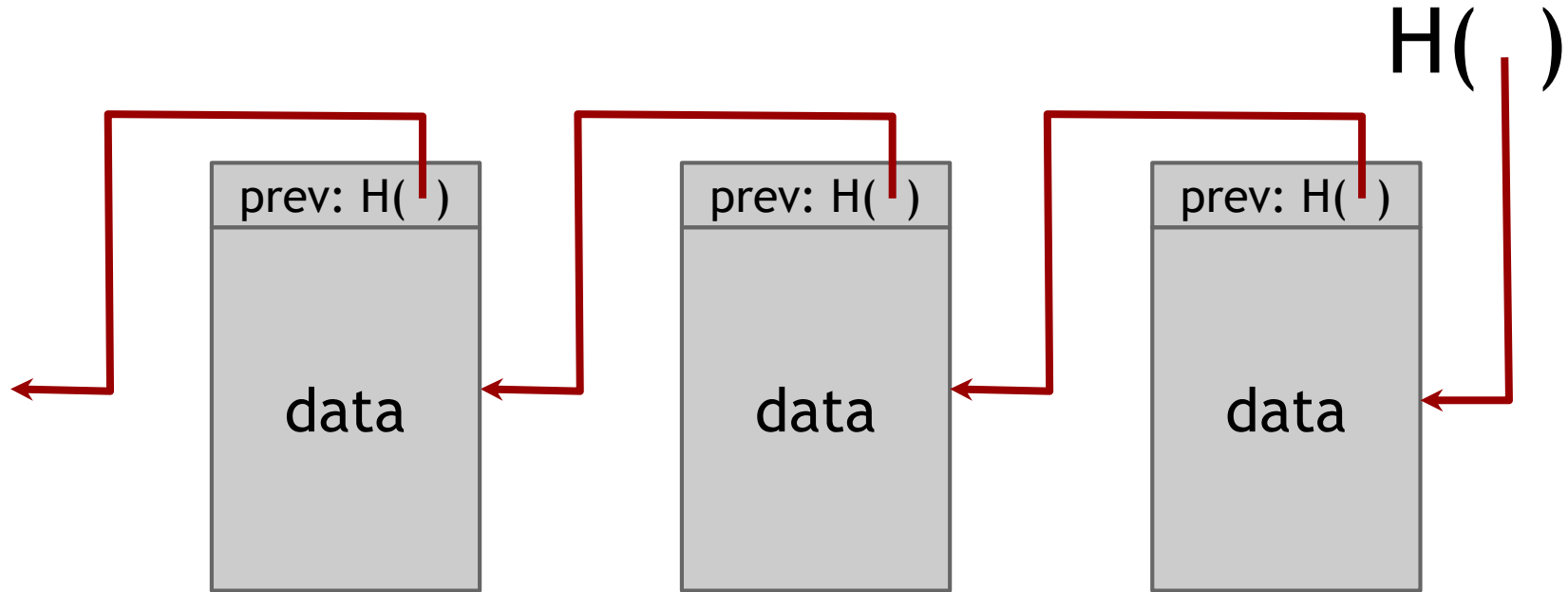
- * ask to get the info back, and
- * verify that it hasn't changed



key idea:

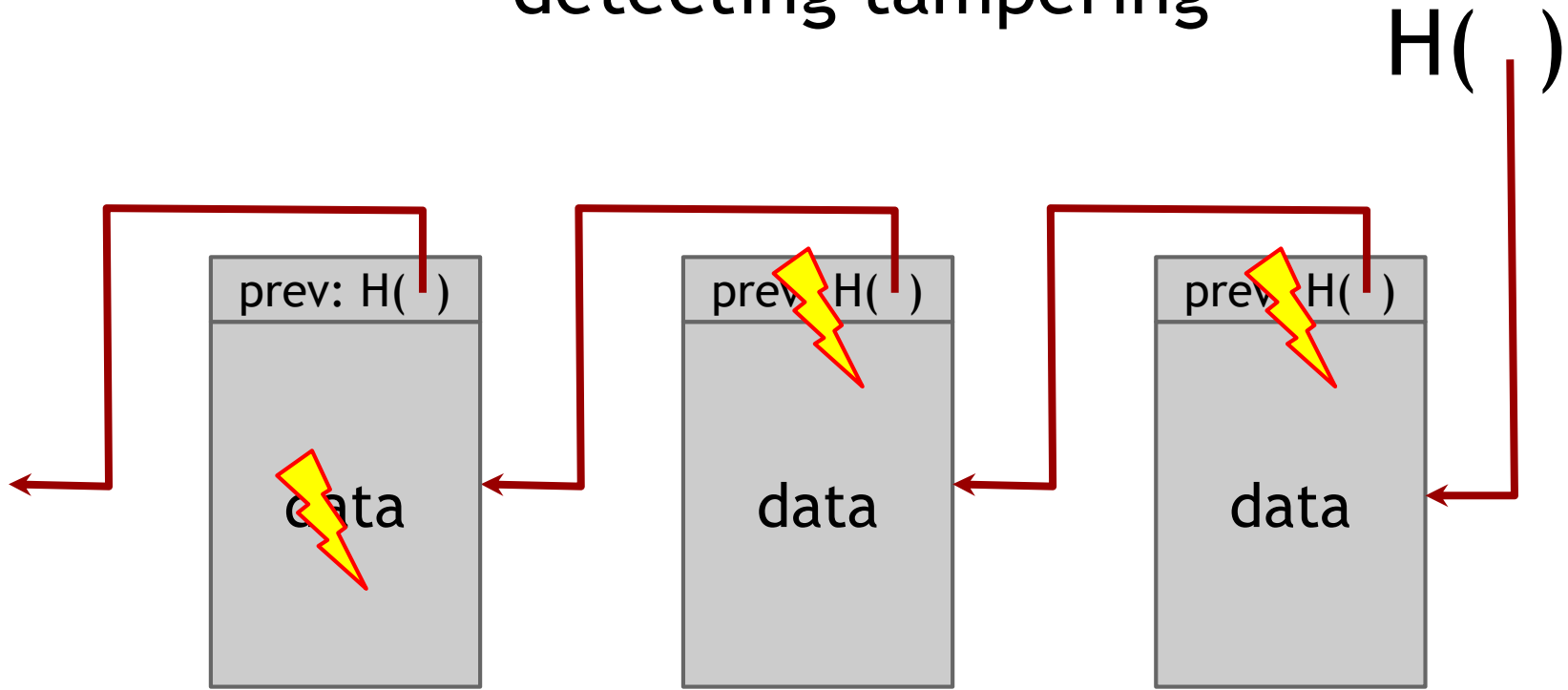
build data structures with hash pointers

linked list with hash pointers = “block chain”



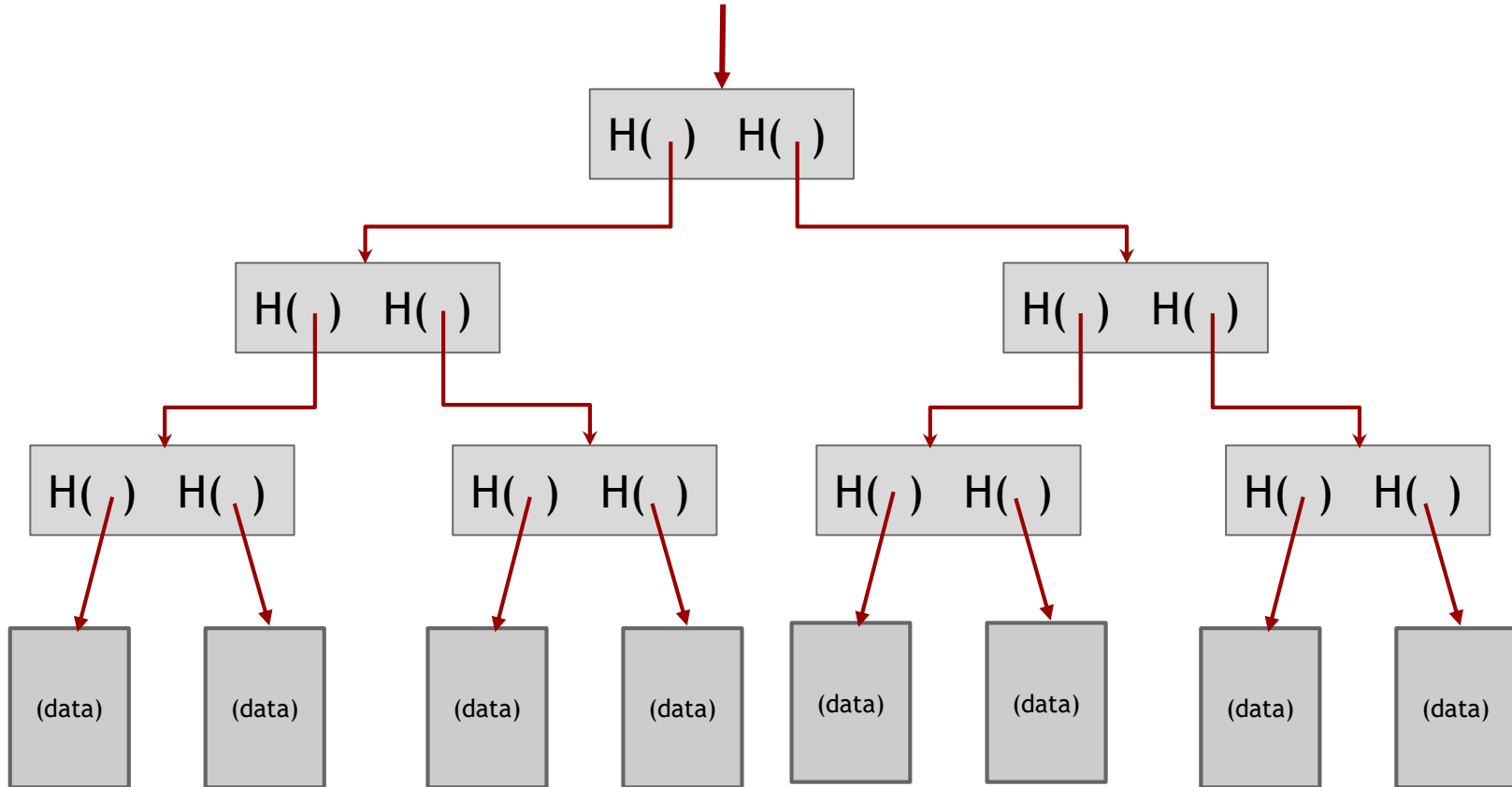
use case: tamper-evident log

detecting tampering

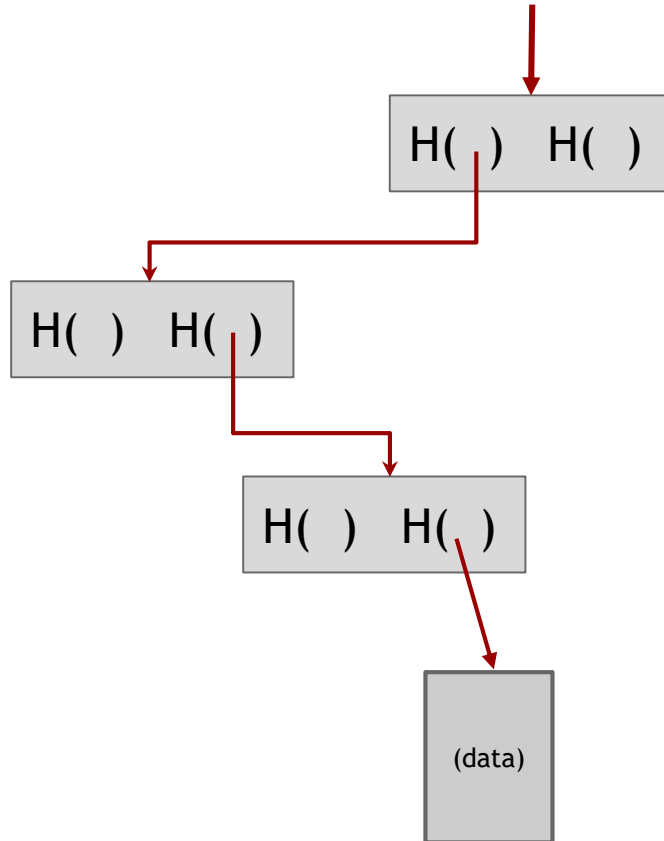


use case: tamper-evident log

binary tree with hash pointers = “Merkle tree”



proving membership in a Merkle tree



show $O(\log n)$ items

More generally ...

can use hash pointers in any pointer-based
data structure that has no cycles



GoofyCoin

Simple Cryptocurrencies

Obvious approach

1. Use public keys as addresses
2. Sign to authorize transfer to new address

New coins created [somehow]

Goofy can create new coins

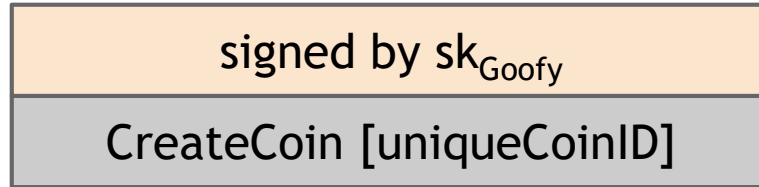
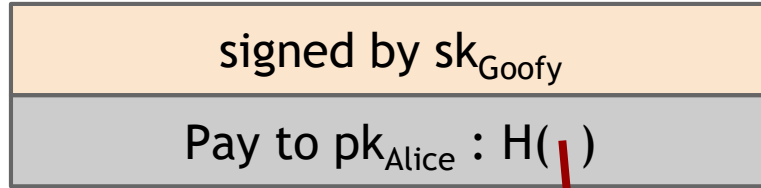
signed by sk_{Goofy}

CreateCoin [uniqueCoinID]

New coins belong to me.



A coin's owner can spend it.



Alice owns it now.



The recipient can pass on the coin again.

signed by sk_{Alice}
Pay to $pk_{\text{Bob}} : H()$

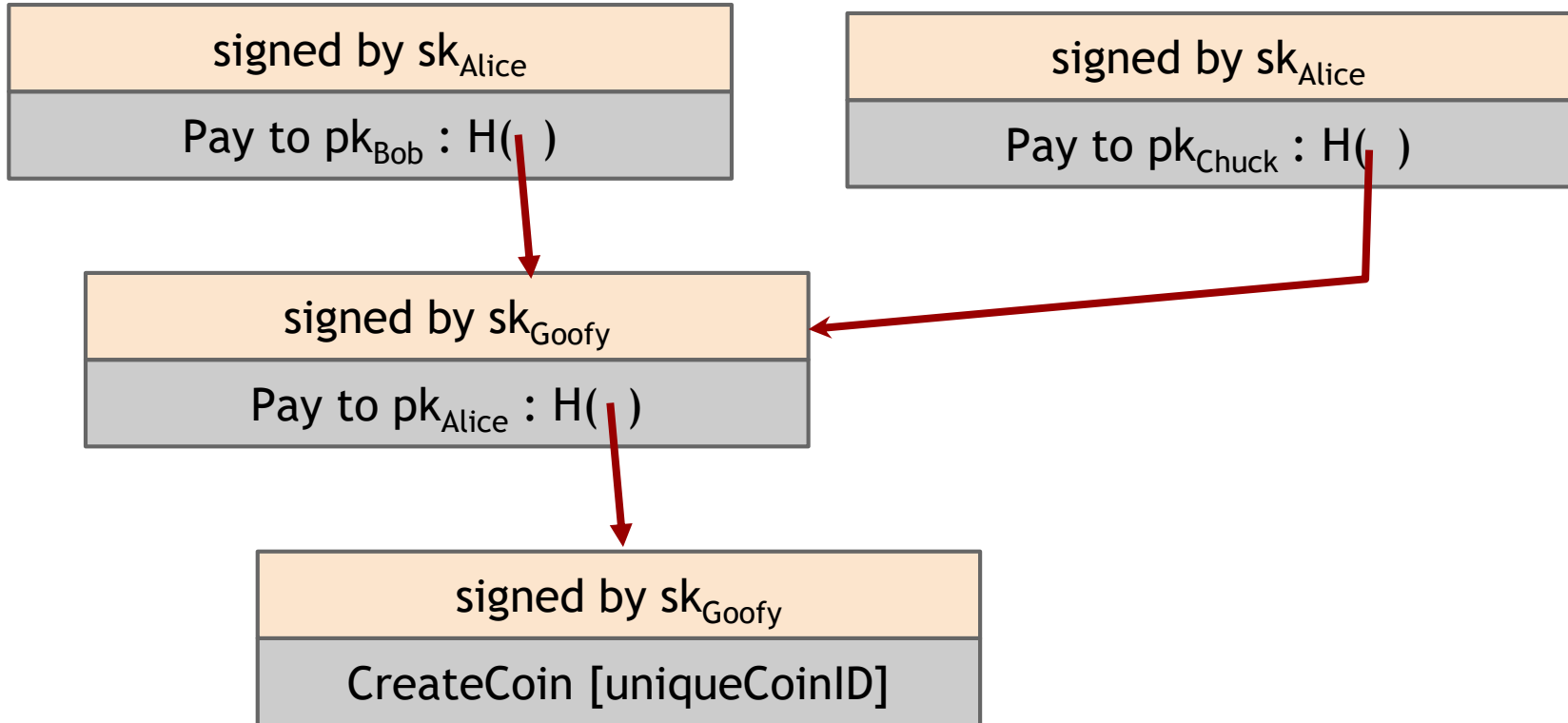
signed by sk_{Goofy}
Pay to $pk_{\text{Alice}} : H()$

signed by sk_{Goofy}
CreateCoin [uniqueCoinID]

Bob owns it now.



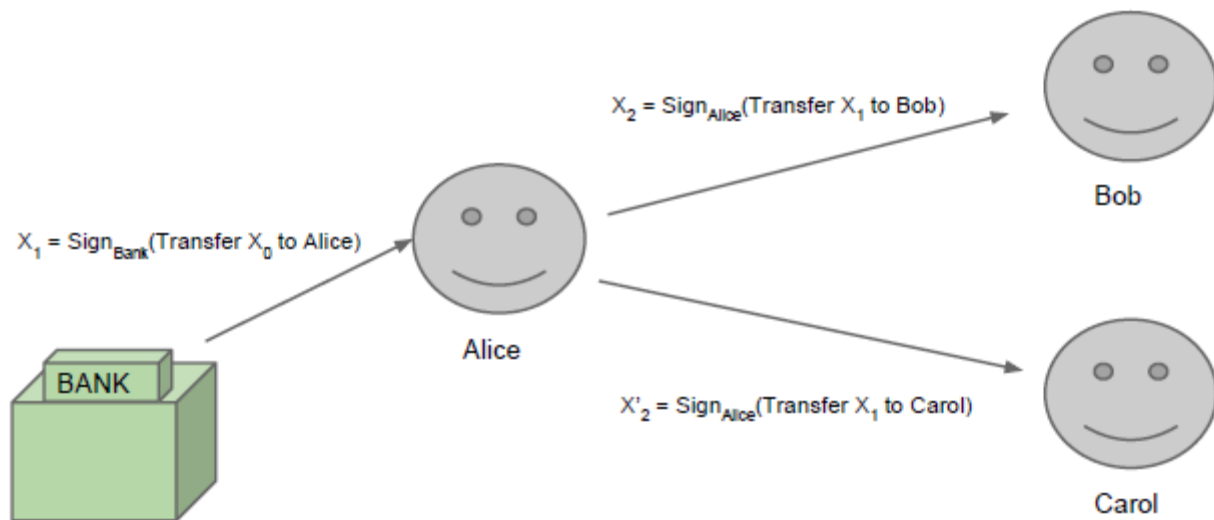
double-spending attack



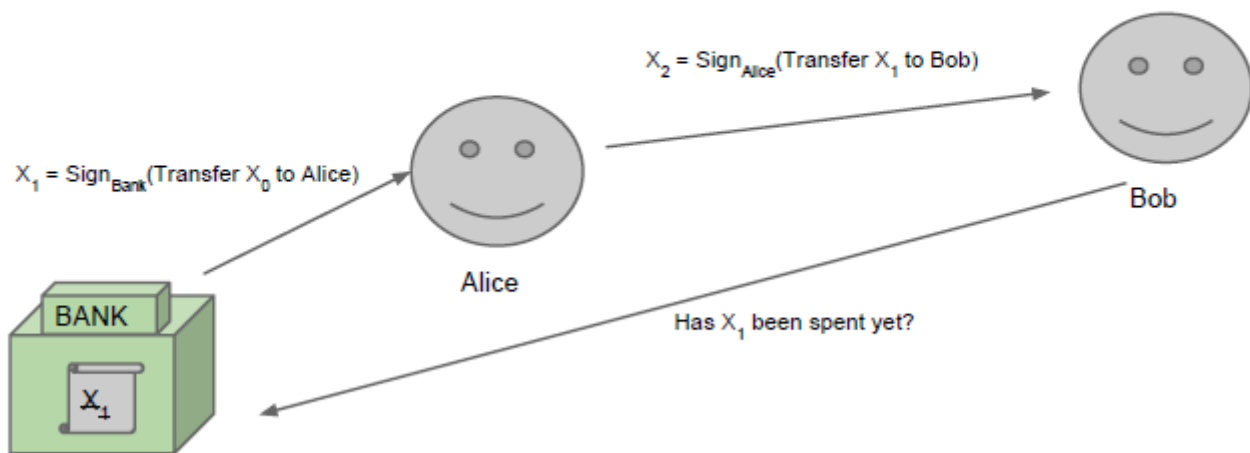
double-spending attack

the main design challenge in digital currency

Double-spends must be prevented



Traditional approach: talk to the issuer





ScroogeCoin

CreateCoins transaction creates new coins

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

← coinID 73(0)

← coinID 73(1)

← coinID 73(2)

Valid, because I said so.



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

transID: 74 type:PayCoins		
consumed coinIDs: 73(1), 42(0), 67(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		

Valid if:

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

Immutable coins

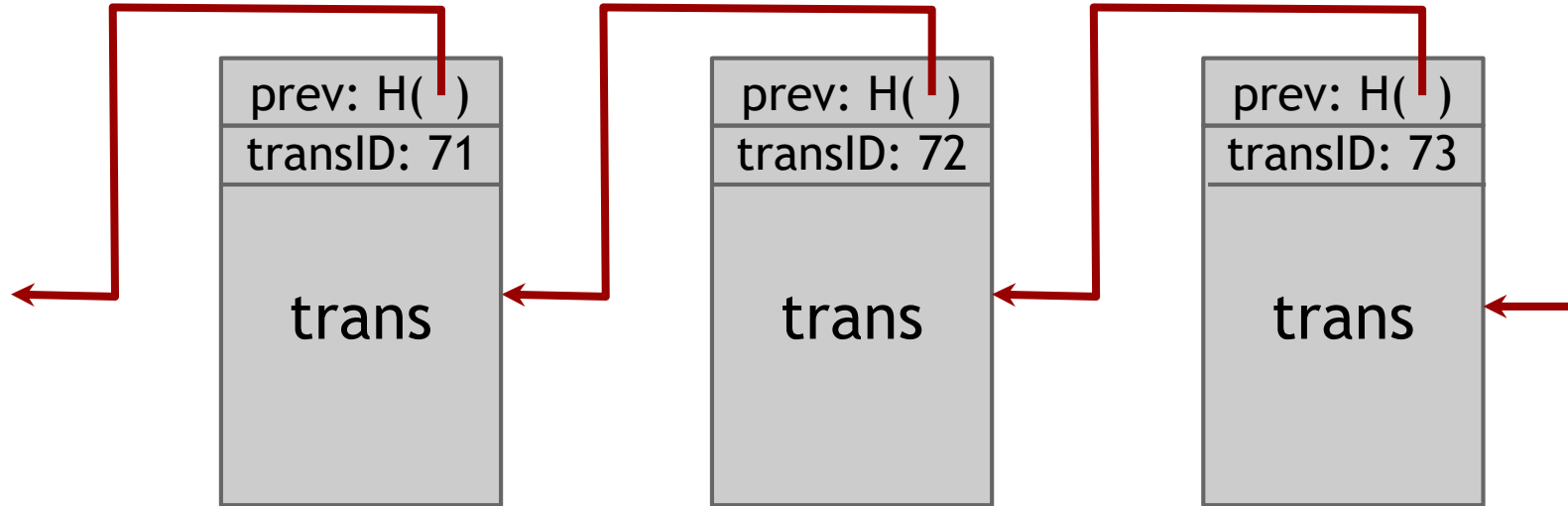
Coins can't be transferred, subdivided, or combined.

But: you can get the same effect by using transactions
to subdivide: create new trans
consume your coin
pay out two new coins to yourself

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



$H()$

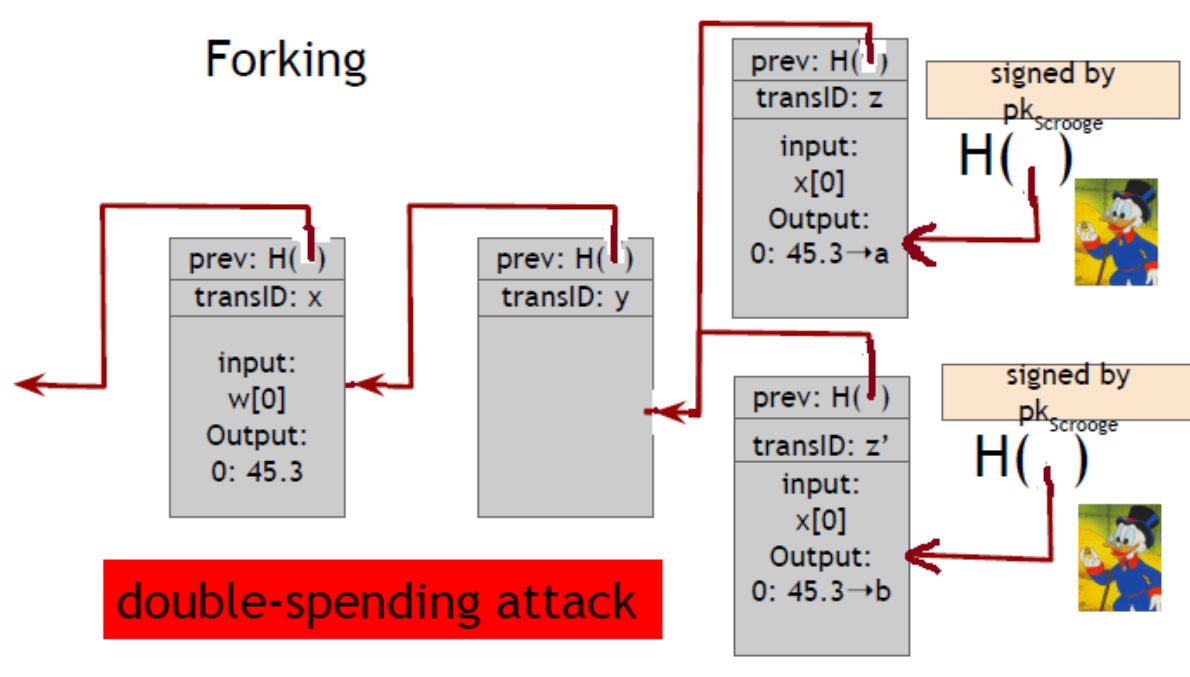


optimization: put multiple transactions in the same block

Don't worry, I'm honest.



Forking



What if Scrooge is malicious?

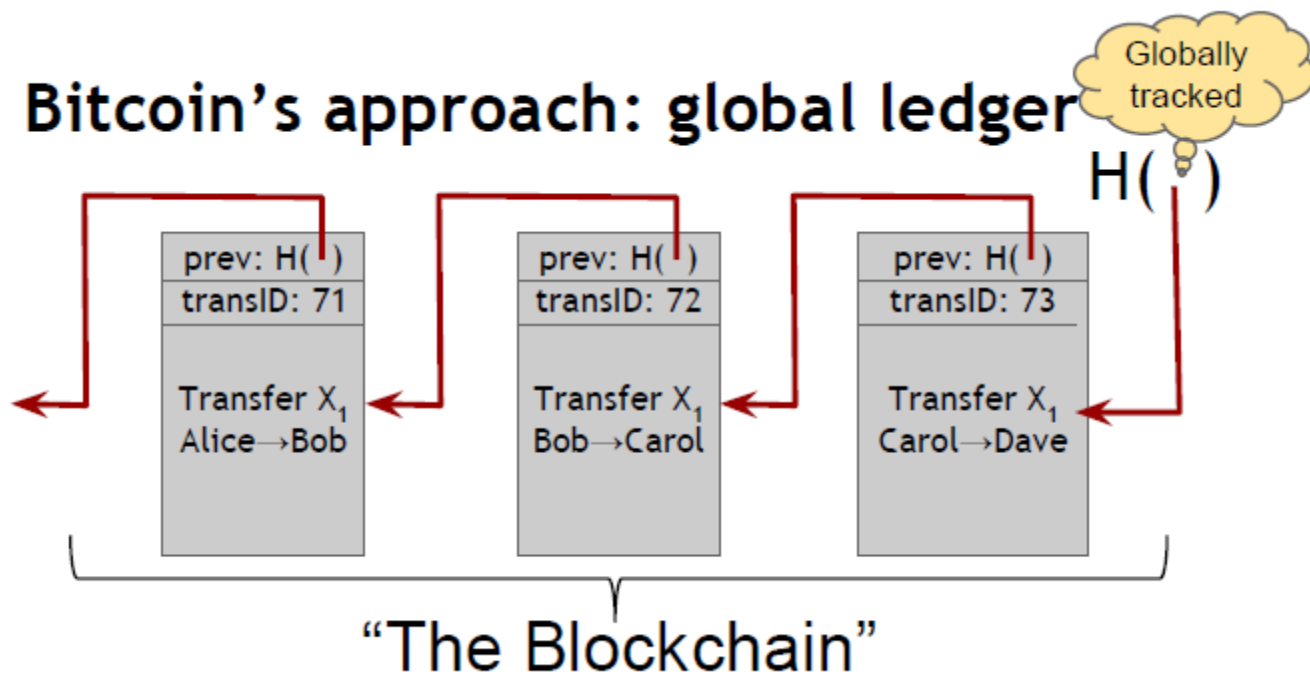
Don't worry, I'm honest.



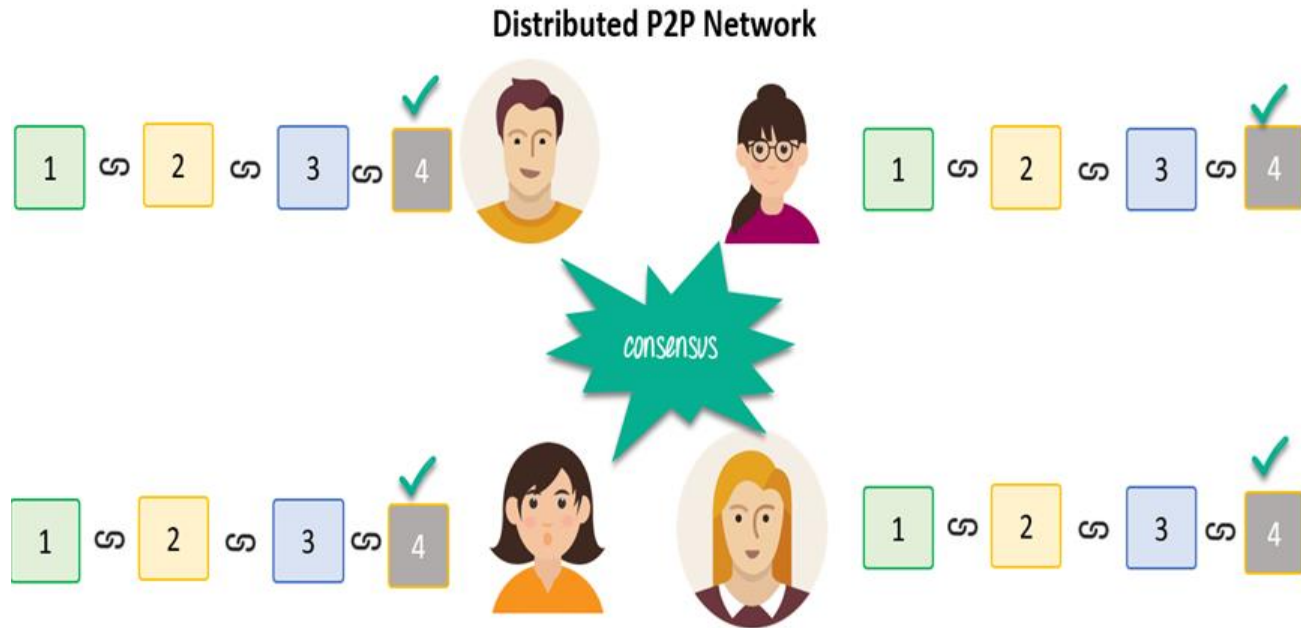
Crucial question:

Can we descroogify the
currency, and operate without
any central, trusted party?

Bitcoin's approach: global ledger



How to achieve consistency?



An Example of Public Ledger from Banking Sectors

Public Ledger
of Alice

Alice: ₹100



Alice
₹ 100



Bob

Alice: ₹100

Public Ledger
of Bob

Public Ledger
of Eve

Alice: ₹100



Eve

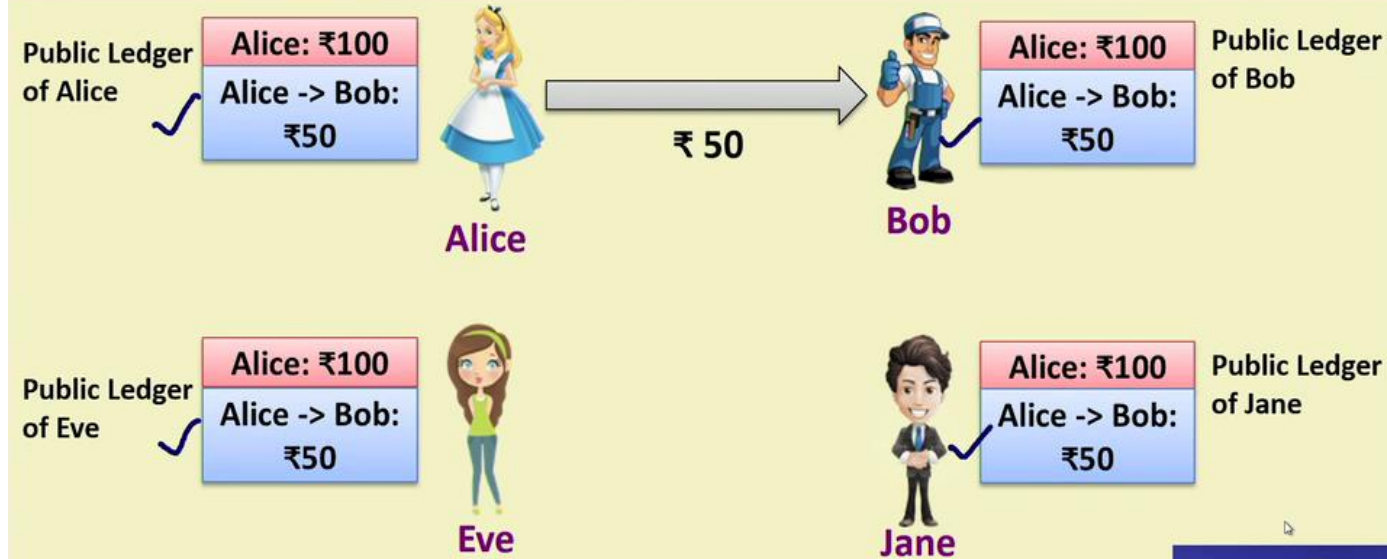


Jane

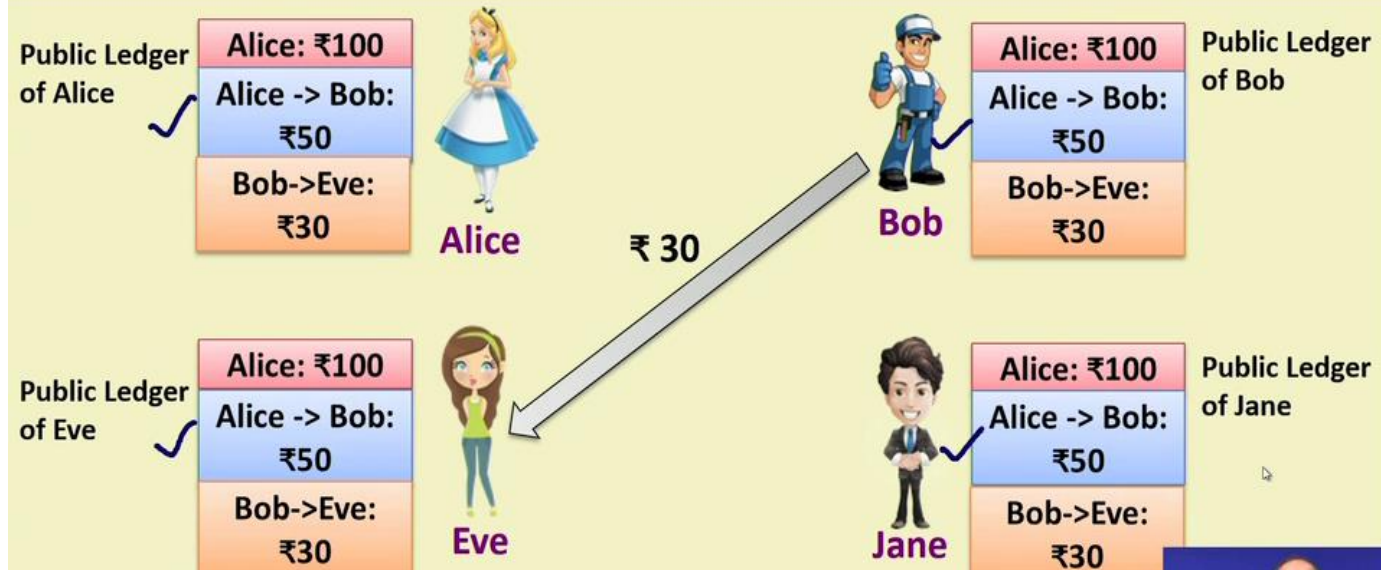
Alice: ₹100

Public Ledger
of Jane

An Example of Public Ledger from Banking Sectors



An Example of Public Ledger from Banking Sectors

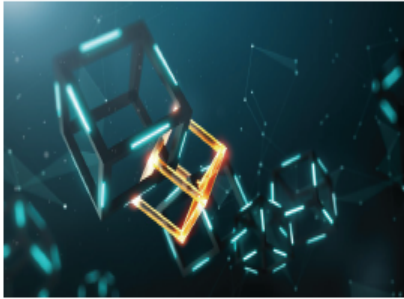


An Example of Public Ledger from Banking Sectors



The path to decentralization

- technology & incentive design



Who determines the
validity of transactions to
be included in the ledger?



Who maintains the ledger
of transactions? (and how?)

All
Participants

Consensus

All
Participants

Bitcoin
Script

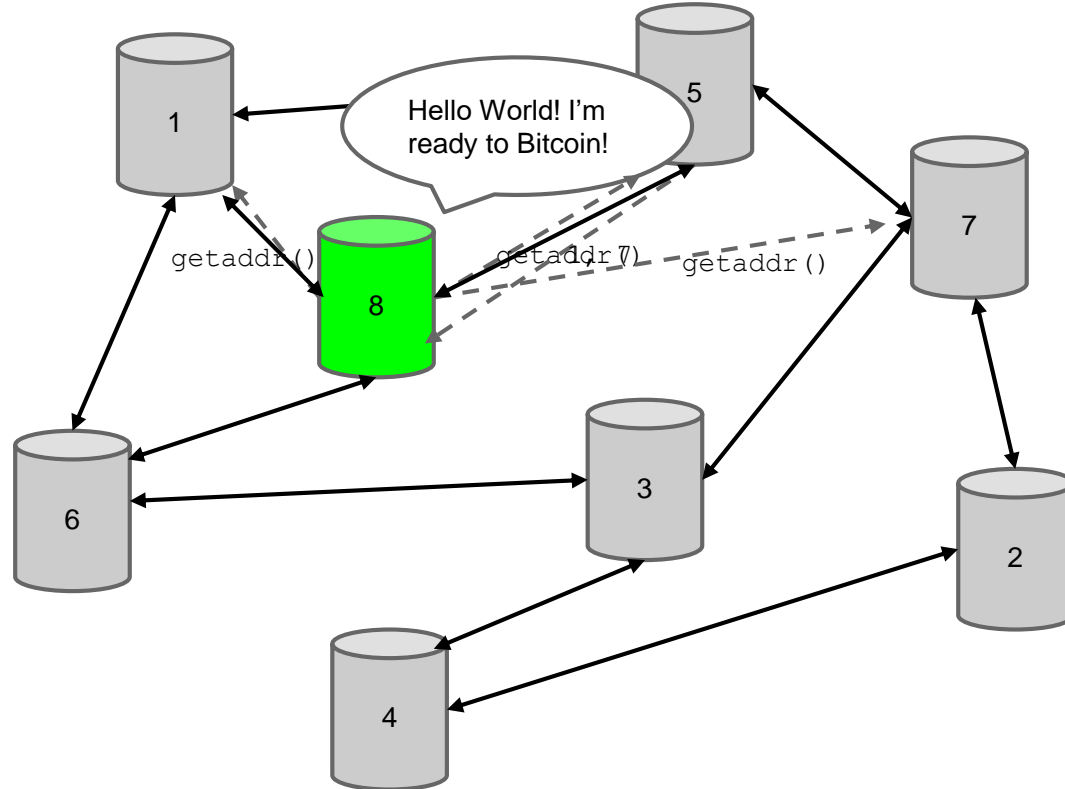
Who creates new Bitcoins?

Reward
for Mining

Bitcoin P2P network

- Ad-hoc protocol (runs on TCP port 8333)
- Ad-hoc network with random topology
- All nodes are equal
- New nodes can join at any time
 - Network Changes over time - dynamic
- No explicit way to leave network
 - Forget non-responding nodes after 3 hr

Joining the Bitcoin P2P network

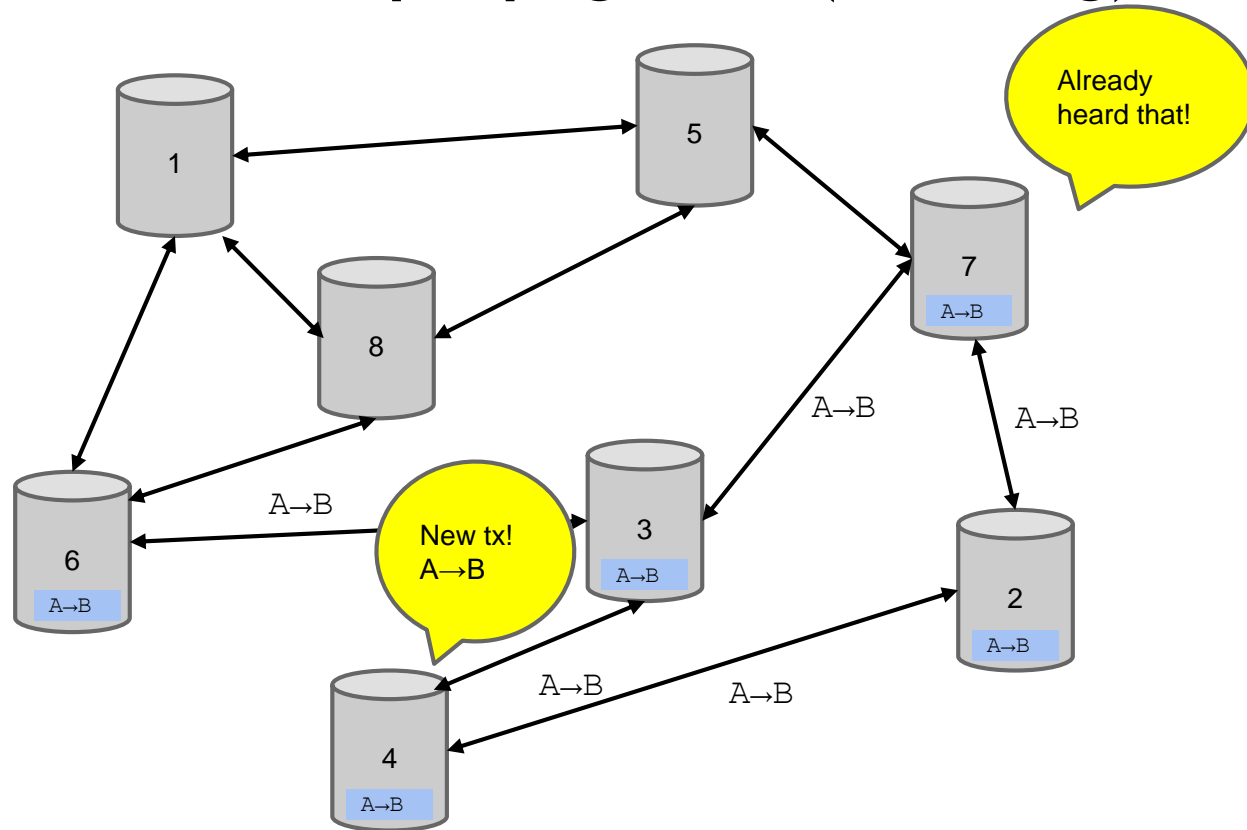


BLOCKCHAIN WORKING PRINCIPLE

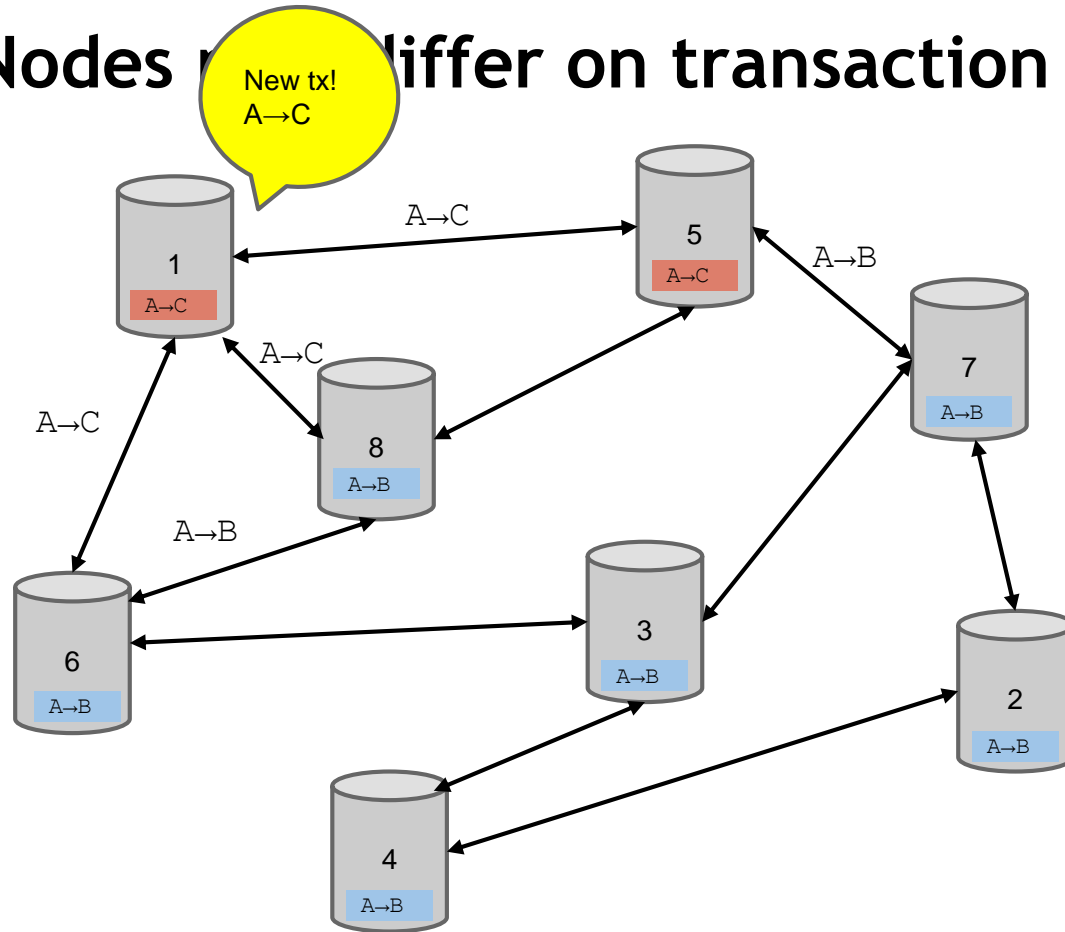


HOW THE
BLOCKCHAIN
WORKS?

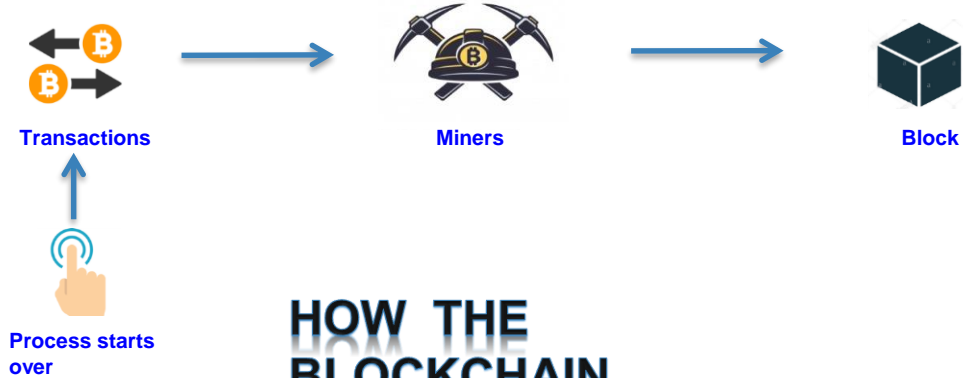
Transaction propagation (flooding)



Nodes differ on transaction pool

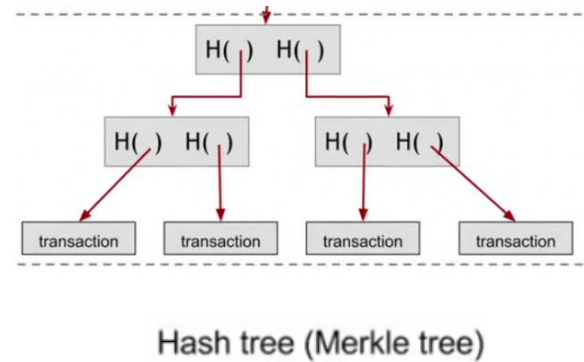
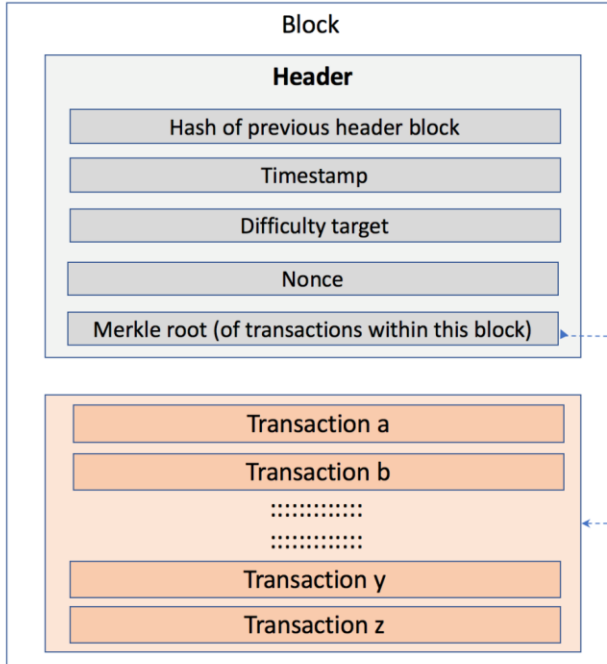


BLOCKCHAIN WORKING PRINCIPLE

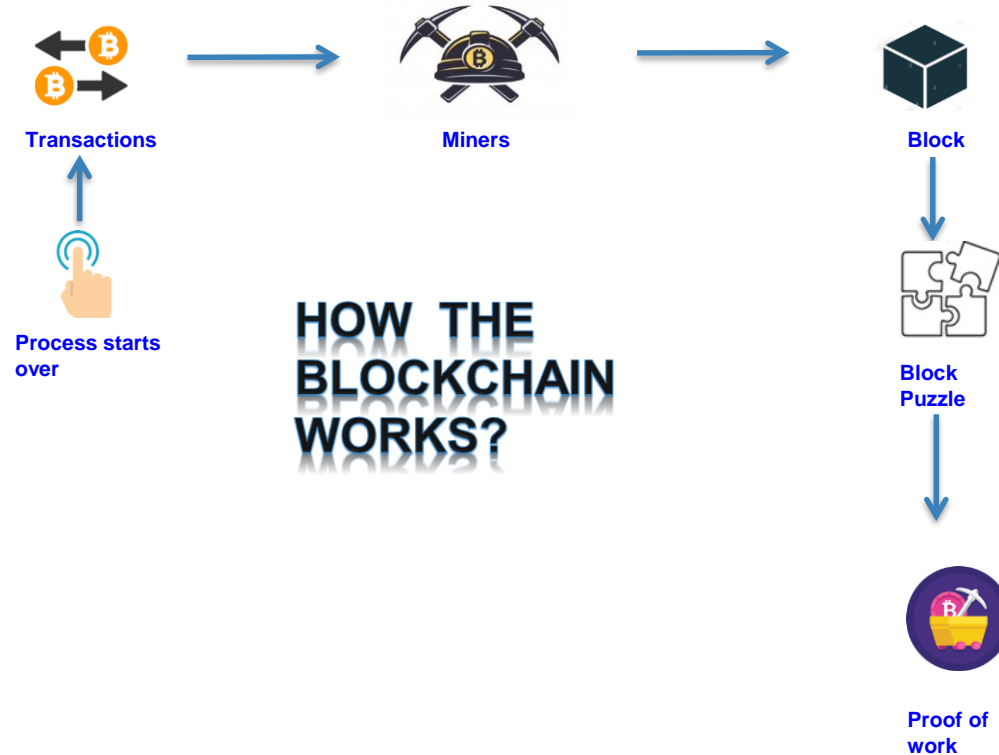


HOW THE
BLOCKCHAIN
WORKS?

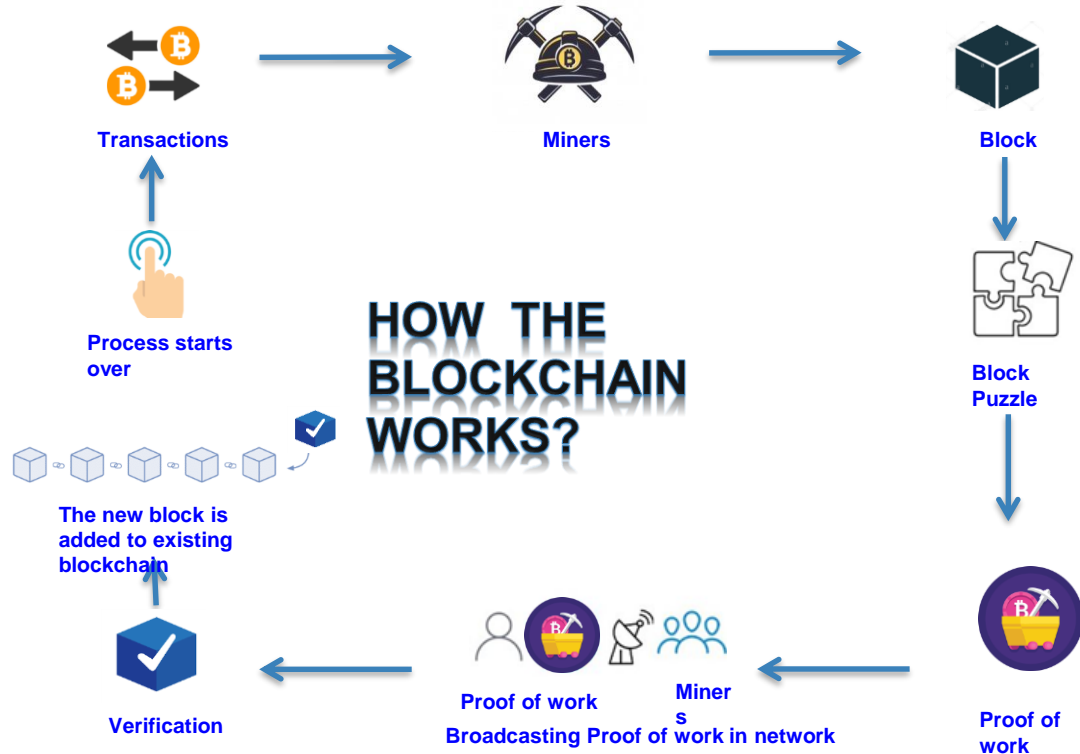
Block Structure



BLOCKCHAIN WORKING PRINCIPLE



BLOCKCHAIN WORKING PRINCIPLE



Mining Bitcoins in 6 easy steps

1. Join the network, listen for transactions
 - a. Validate all proposed transactions
2. Miners will assemble them into new blocks and solve puzzle
3. On success, broadcast the new blocks
4. Listen for new blocks, maintain block chain
 - a. When a new block is proposed, validate it
5. Find the nonce to make your block valid
6. Hope everybody
7. Profit!

Rewards and Transaction Fees!