



Blockchain foundations

Module 1 (part 4)

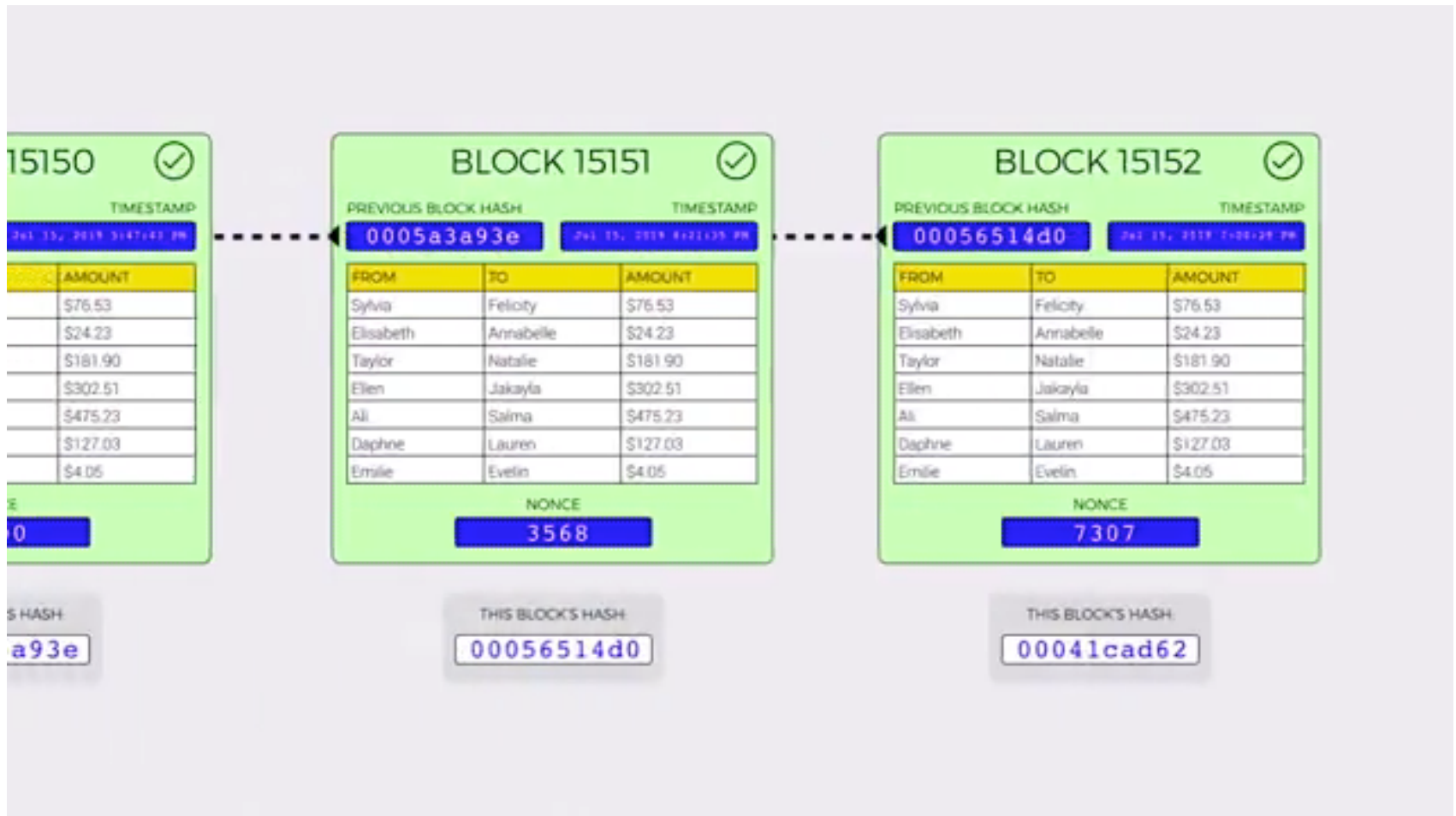
Part 4

Anatomy of a Block

[Source from ConsenSys]

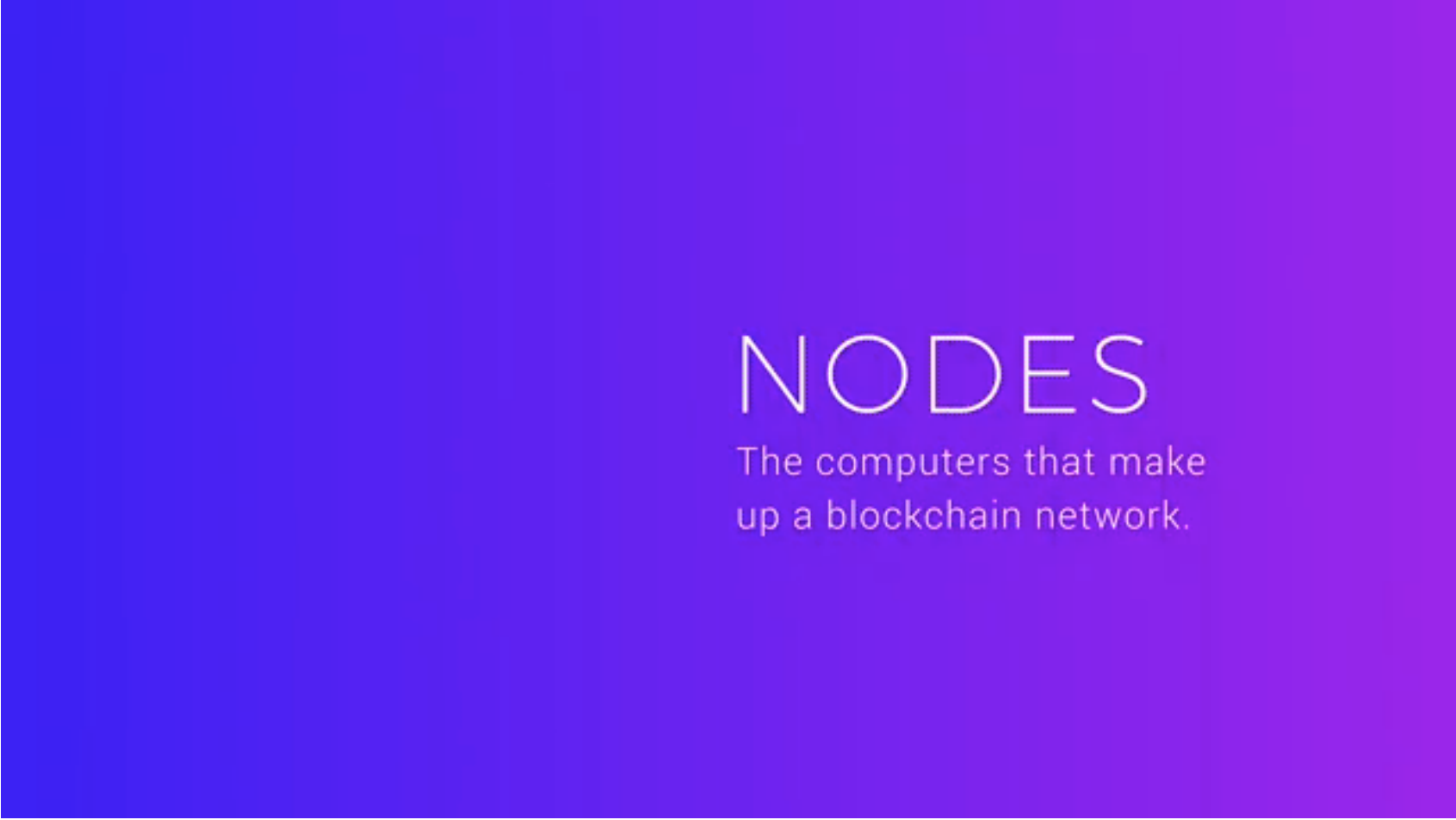


The Chain of Blocks



Nodes and Networks

[Source from ConsenSys]



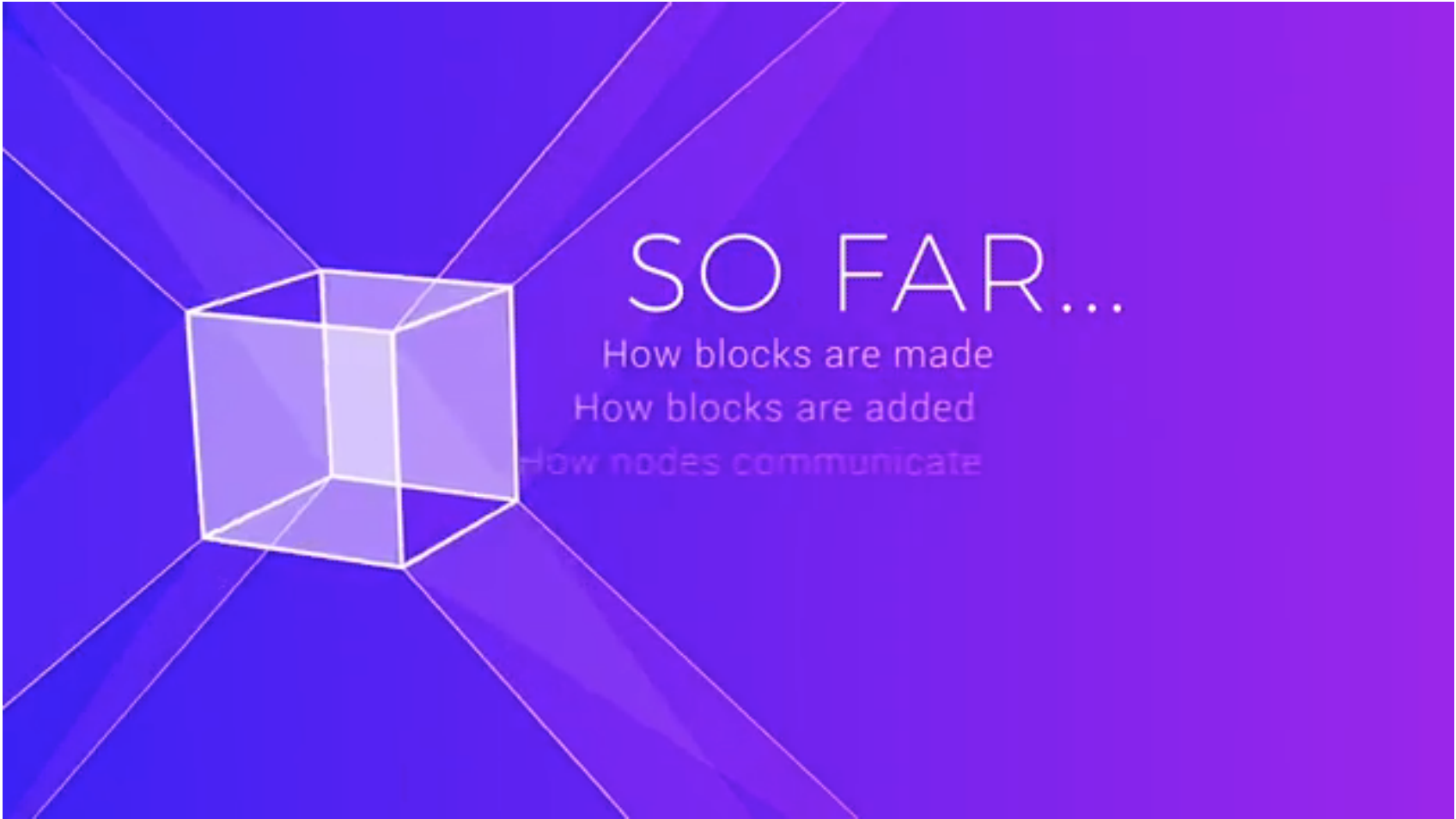
NODES

The computers that make
up a blockchain network.

Consensus Mechanisms/ trust frameworks

- Consensus, as a concept, is fundamental to any system where more than one entity is participating. As the Ethereum blockchain aims to include any participant that runs a node, there is a need to quickly reach agreement on whether to accept a new block as part of the chain when it is **mined**. Without the ability to reach this agreement, the blockchain can not create new blocks. A consensus mechanism is a set of rules that allows multiple machines that are connected together to work together while tolerating some machines providing incorrect data or failing completely. These properties are called fault-tolerance and resilience. A blockchain needs a consensus mechanism to ensure there are a set of rules for **creating and accepting each newly created block**.

[Source from ConsenSys]



Cryptocurrency

- **Coin vs. token**
- **IPO vs. ICO**
- **STO?**

More reading [link](#).



U.S. SECURITIES AND EXCHANGE COMMISSION

COMPANY FILINGS | MORE SEARCH OPTIONS

ABOUT | DIVISIONS & OFFICES | ENFORCEMENT | REGULATION | EDUCATION | FILINGS | NEWS

SEC Spotlight

2017 Broker-Dealer
Compliance

Advisory Committee on
Small and Emerging
Companies

Affinity Fraud

Crowdfunding

Cybersecurity

Disclosure Effectiveness

Enforcement
Cooperation Initiative

Initial Coin Offerings (ICOs)



Companies and individuals are increasingly considering initial coin offerings (ICOs) as a way to raise capital or participate in investment opportunities. While these digital assets and the technology behind them may present a new and efficient means for carrying out financial transactions, they also bring increased risk of fraud and manipulation because the markets for these assets are less regulated than traditional capital markets. That's why we are providing this information about the three "Rs" of ICOs: Risks, Rewards and Responsibilities.

<https://www.sec.gov/ICO>

[Source from ConsenSys]



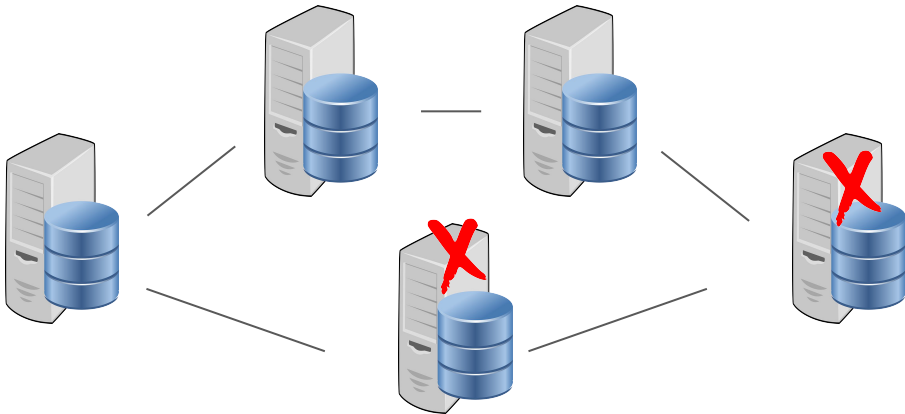
Wallets, Exchanges & Transactions (important)

[Source from ConsenSys]



Permissioned Blockchains

AKA “Consortium blockchain”
Nodes are run by well-known,
mostly trusted entities



Public Blockchains

Open for participation by
anyone



Different blockchain platforms

- Public blockchains (a.k.a. permissionless blockchains)
 - Example of implementation: Bitcoin, Ethereum
- Private blockchains (a.k.a. permissioned blockchains)
 - Example of implementation: Hyperledger Fabric

More reading [link](#).

Question

If you have determined that you need a blockchain, but do not want the transactions to be publicly visible, what sort of blockchain could you use? Select all that apply.

- ☐ Public blockchain
- ☐ Private blockchain
- ☐ Consortium blockchain

When to use a blockchain

[Source from ConsenSys]



More reading [link](#).

Bitcoin

- Consensus, distributed ledgers
- Transactions, proof-of-work

How a Bitcoin transaction works

Bob, an online merchant, decides to begin accepting bitcoins as payment. Alice, a buyer, has bitcoins and wants to purchase merchandise from Bob.

WALLETS AND ADDRESSES



Bob and Alice both have Bitcoin "wallets" on their computers.



Wallets are files that provide access to multiple Bitcoin addresses.



An address is a string of letters and numbers, such as 1HULMwZEPkjEPeCh43BeKJLybLCWrfDpN.

CREATING A NEW ADDRESS

Bob creates a new Bitcoin address for Alice to send her payment to.



Each address has its own balance of bitcoins.

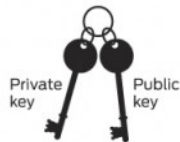
SUBMITTING A PAYMENT



Alice tells her Bitcoin client that she'd like to transfer the purchase amount to Bob's address.

Public Key Cryptography 101

When Bob creates a new address, what he's really doing is generating a "cryptographic key pair," composed of a private key and a public key. If you sign a message with a private key (which only you know), it can be verified by using the matching public key (which is known to anyone). Bob's new Bitcoin address represents a unique public key, and the corresponding private key is stored in his wallet. The public key allows anyone to verify that a message signed with the private key is valid.



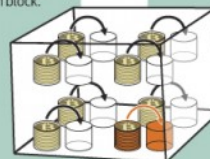
It's tempting to think of addresses as bank accounts, but they work a bit differently. Bitcoin users can create as many addresses as they wish and in fact are encouraged to create a new one for every new transaction to increase privacy. So long as no one knows which addresses are Alice's, her anonymity is protected.

VERIFYING THE TRANSACTION

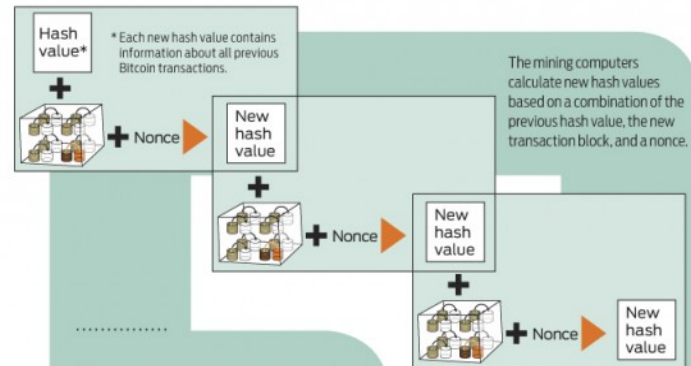


Gary, Garth, and Glenn are Bitcoin miners.

Their computers bundle the transactions of the past 10 minutes into a new "transaction block."



The miners' computers are set up to calculate cryptographic hash functions.



Cryptographic Hashes

Cryptographic hash functions transform a collection of data into an alphanumeric string with a fixed length, called a hash value. Even tiny changes in the original data drastically change the resulting hash value. And it's essentially impossible to predict which initial data set will create a specific hash value.

The root of all evil	6d0a 1899 086a... (56 more characters)
The root of all evil	486c 6be4 6dde...
The root of all evil	b8db 7ee9 8392...

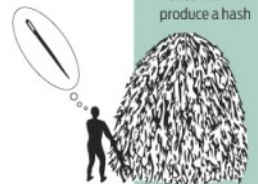
Nonces

To create different hash values from the same data, Bitcoin uses "nonces." A nonce is just a random number that's added to data prior to hashing. Changing the nonce results in a wildly different hash value.

The root of all evil ??? → 0000 0000 0000 ...

Creating hashes is computationally trivial, but the Bitcoin system requires that the new hash value have a particular form—specifically, it must start with a certain number of zeros.

The miners have no way to predict which nonce will produce a hash



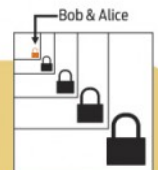
value with the required number of leading zeros. So they're forced to generate many hashes with different nonces until they happen upon one that works.

Each block includes a "coinbase" transaction that pays out 50 bitcoins to the winning miner—in this case, Gary. A new address is created in Gary's wallet with a balance of newly minted bitcoins.



TRANSACTION VERIFIED

As time goes on, Alice's transfer to Bob gets buried beneath other, more recent transactions. For anyone to modify the details, he would have to redo the work that Gary did—because any changes require a completely different winning nonce—and then redo the work of all the subsequent miners. Such a feat is nearly impossible.



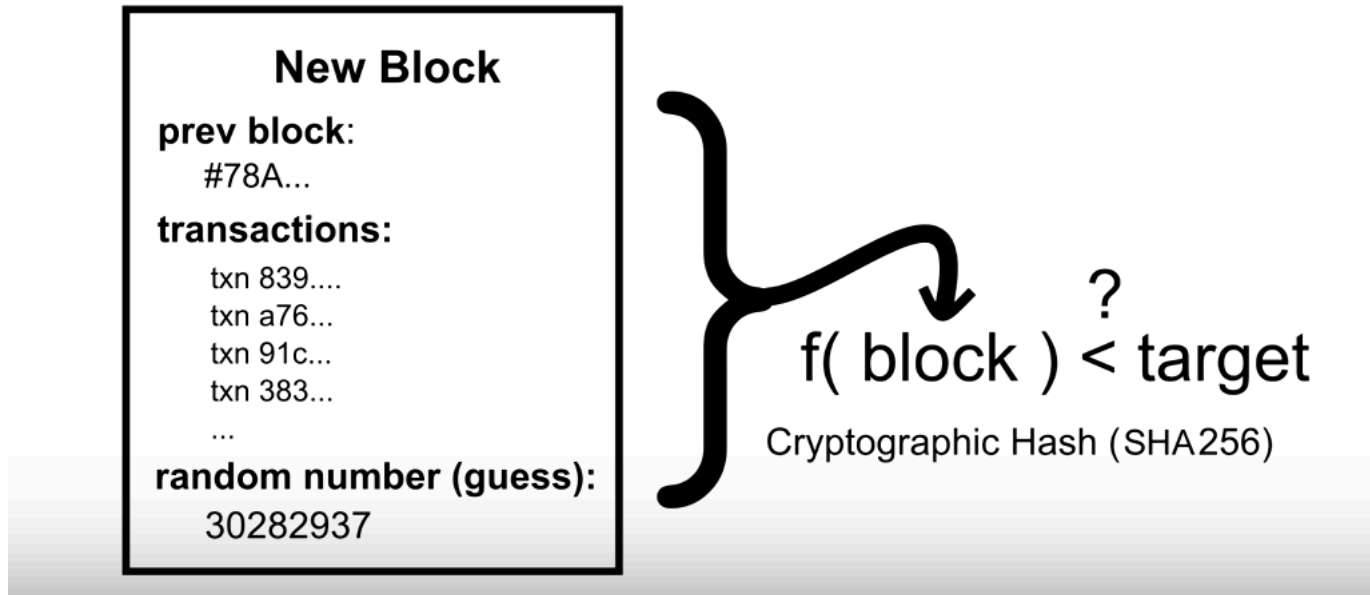
Private key → Public key

Alice's wallet holds the private key for each of her addresses. The Bitcoin client signs her transaction request with the private key of the address she's transferring bitcoins from.

Anyone on the network can now use the public key to verify that the transaction request is actually coming from the legitimate account owner.

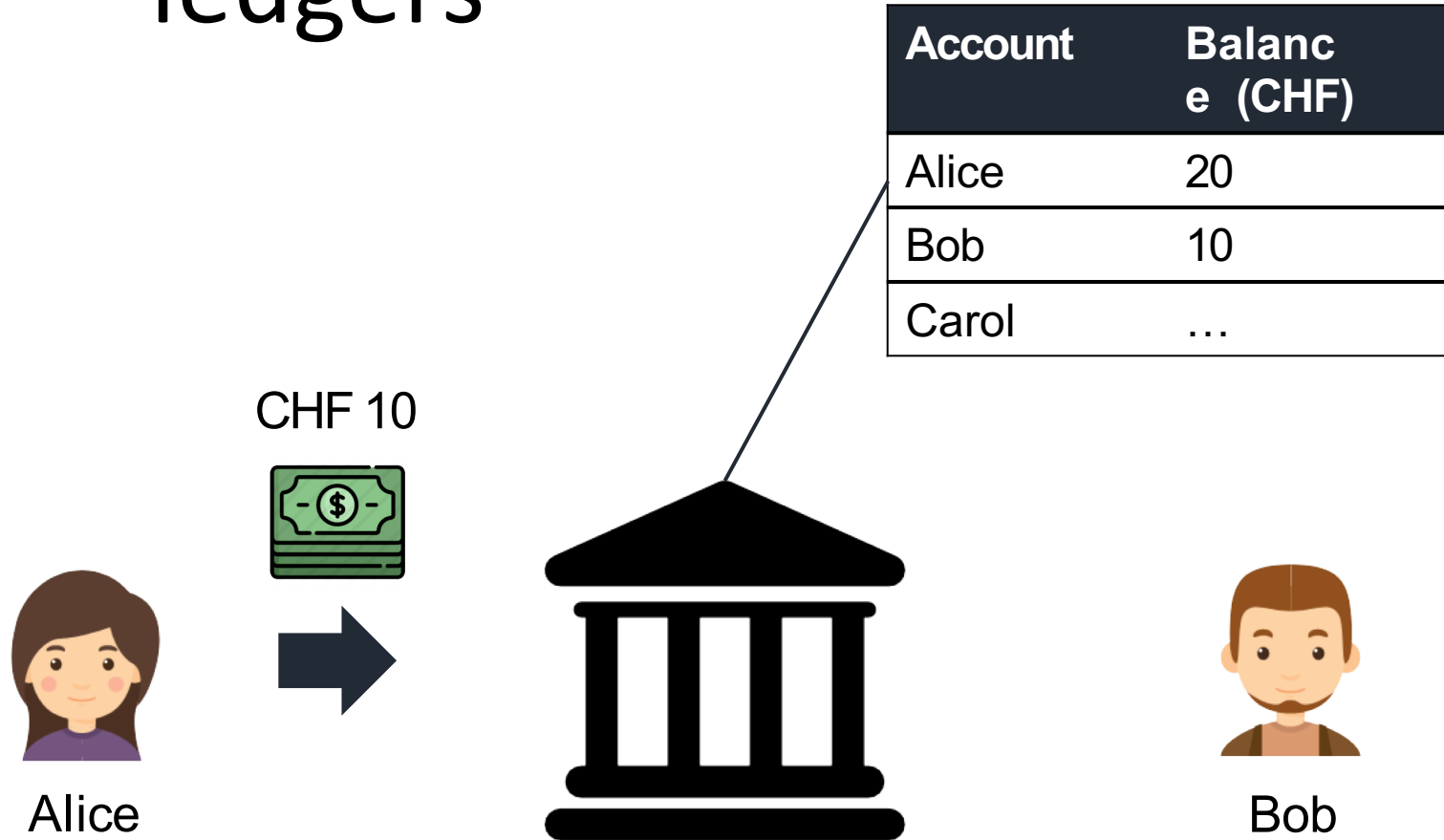
What is the "cryptographic puzzle"?

Block Puzzle

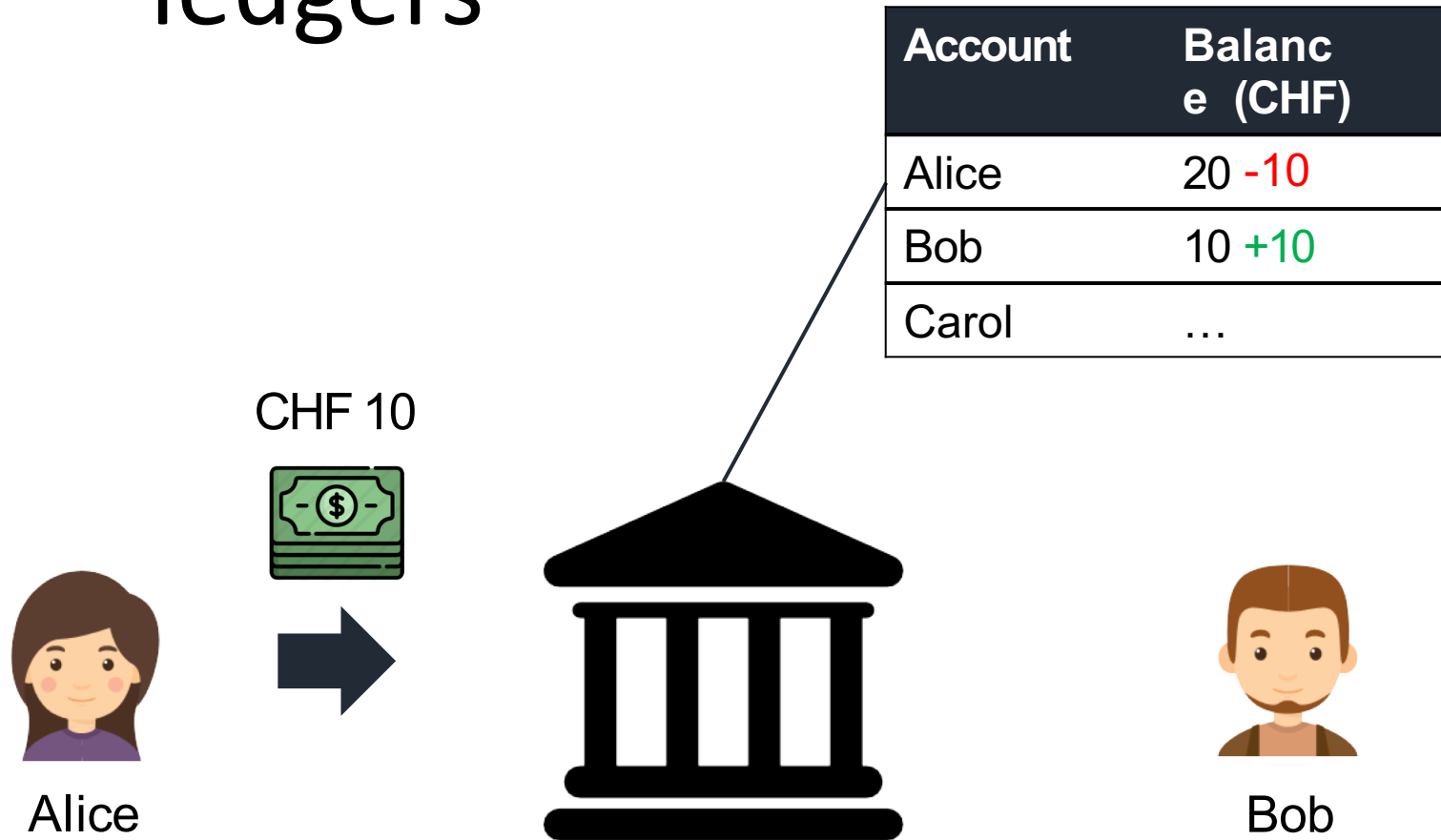


Miners will create the hash of the whole block then compare it with difficulty target....

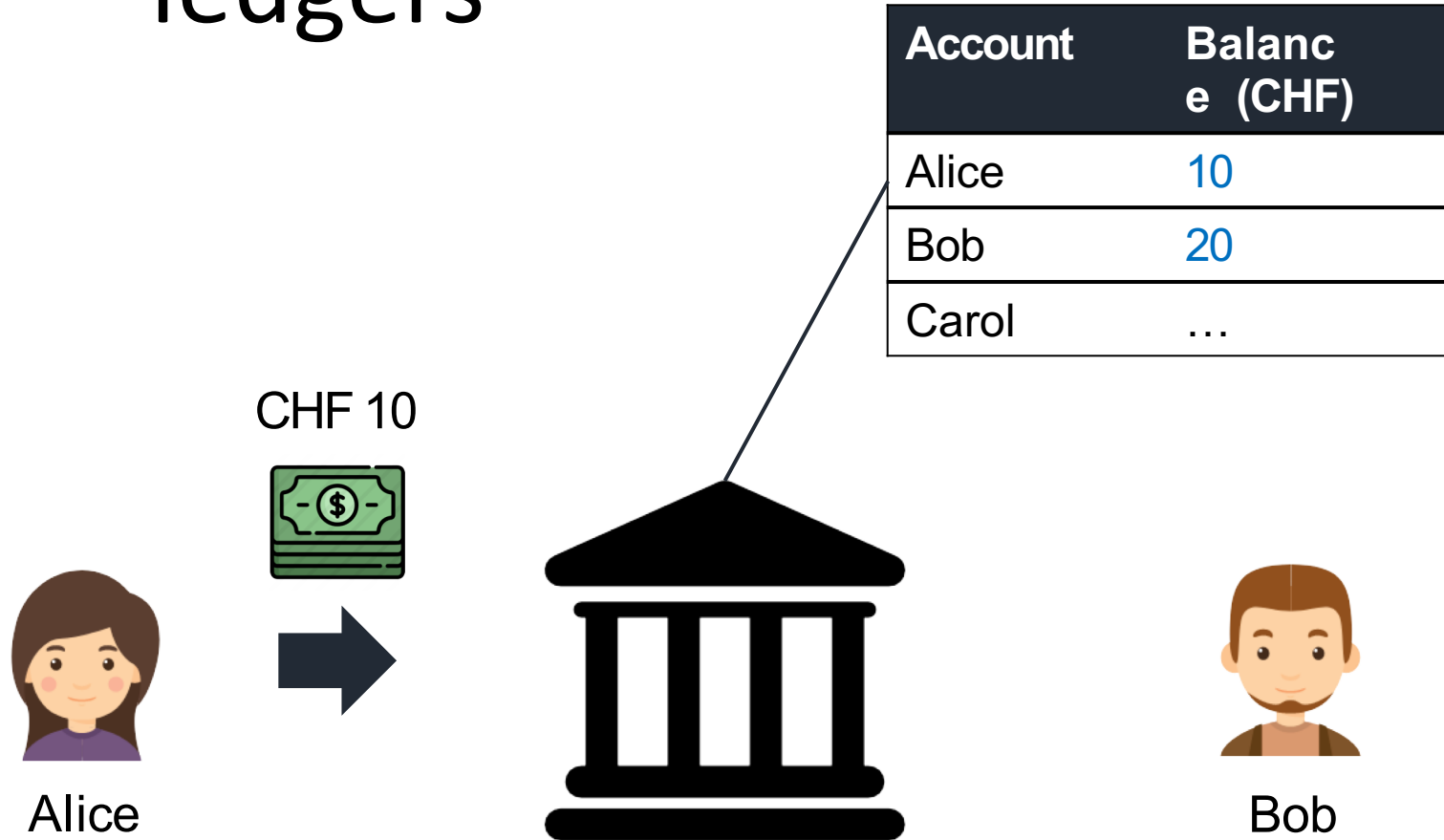
Centralized ledgers



Centralized ledgers



Centralized ledgers



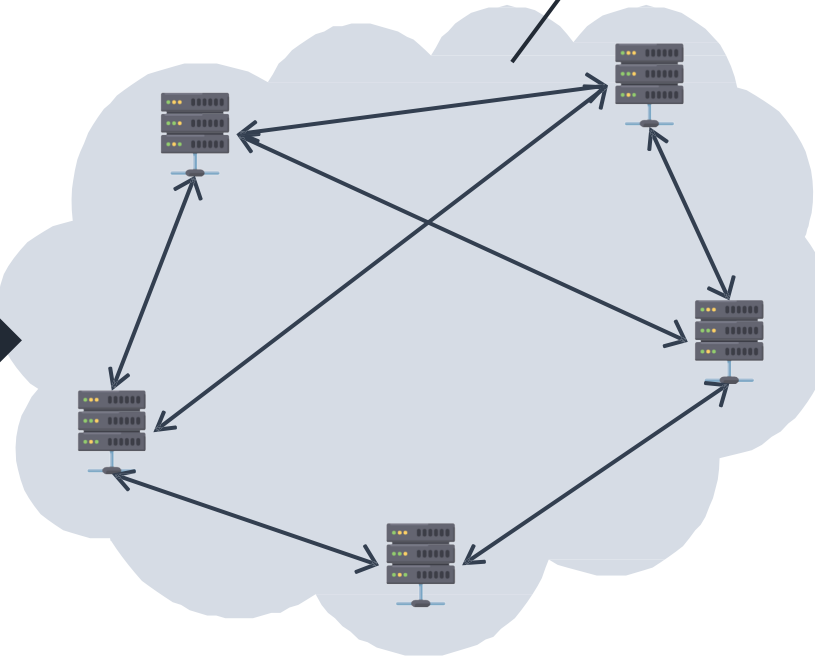
Distributed ledgers

User	Debit	Credit
Alice	20	
Bob	10	

Send 10 coins
to Bob



Alice



Bob

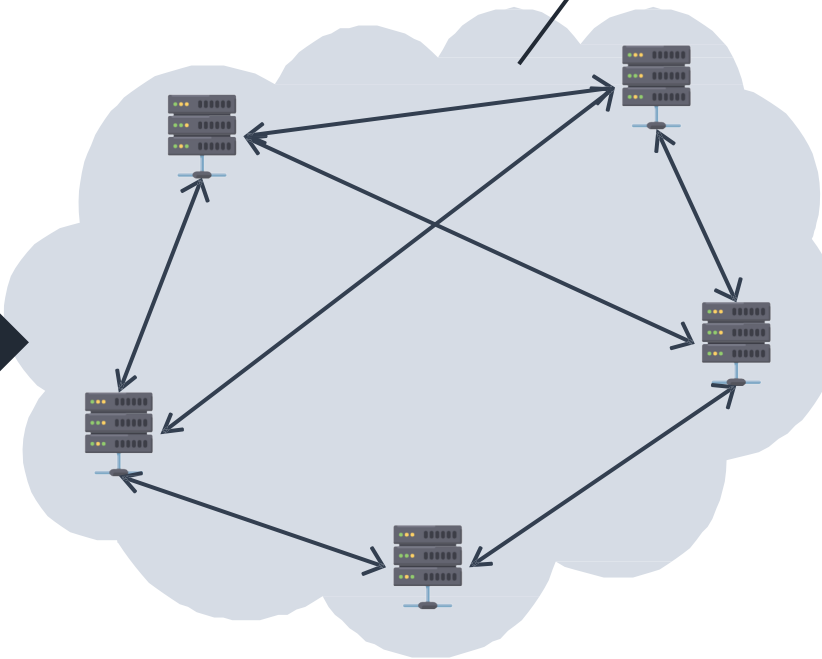
Distributed ledgers

User	Debit	Credit
Alice	20	
Bob	10	
Alice		10
Bob	10	

Send 10 coins
to Bob



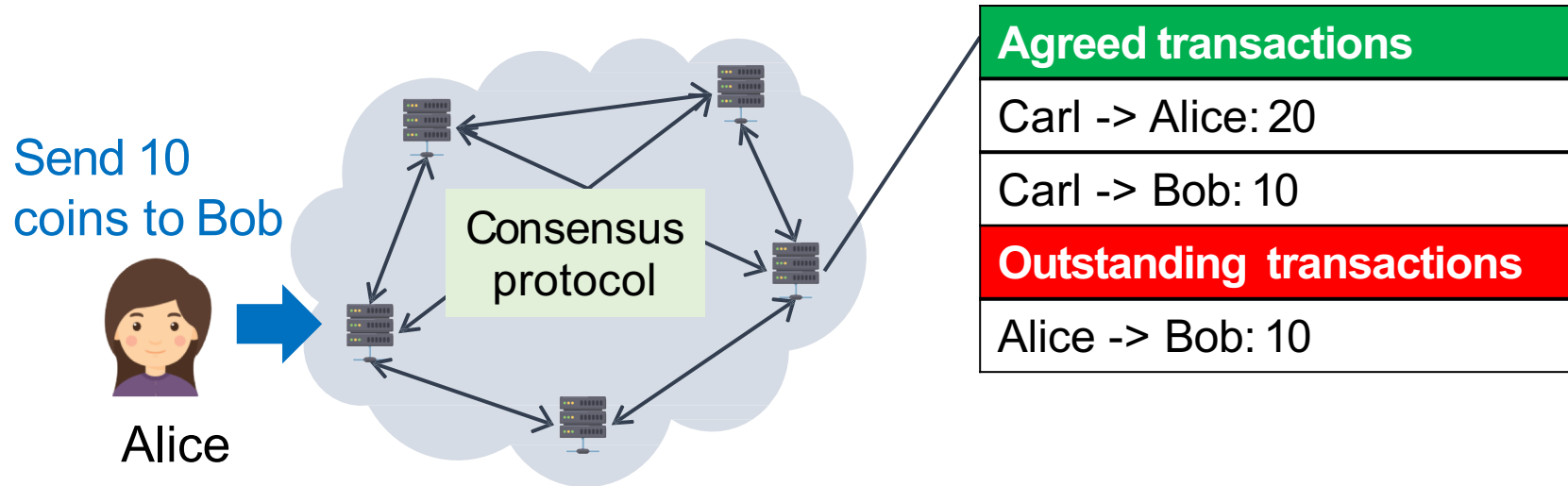
Alice



Bob

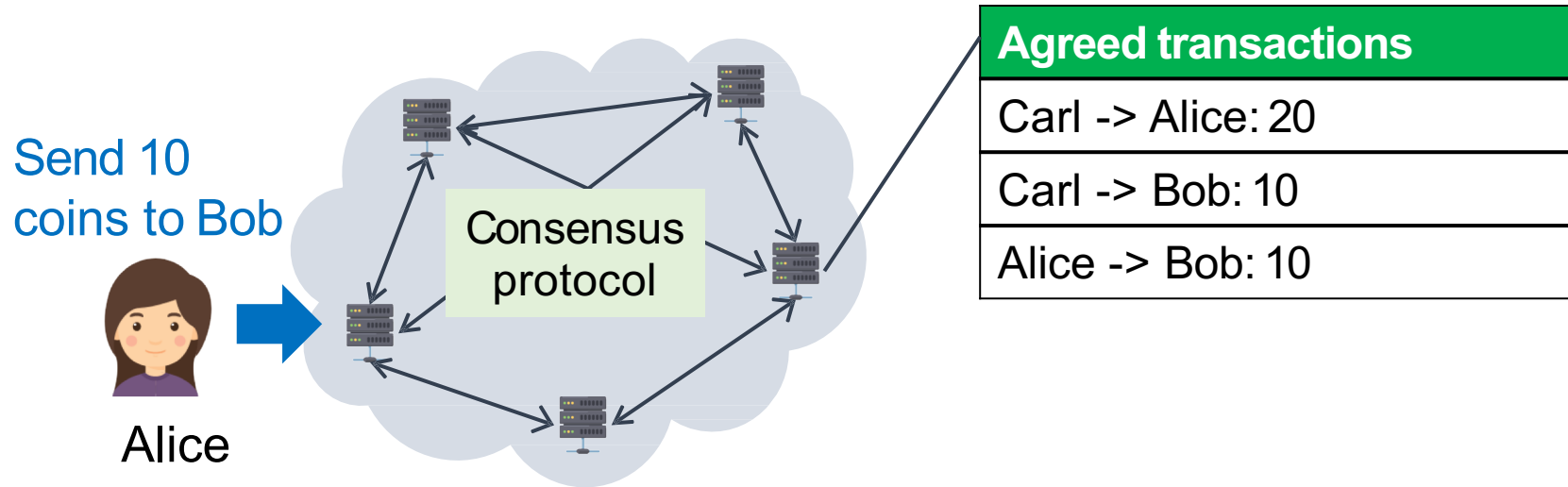
Distributed consensus: all nodes must see the same state of the ledger

Distributed ledger via consensus



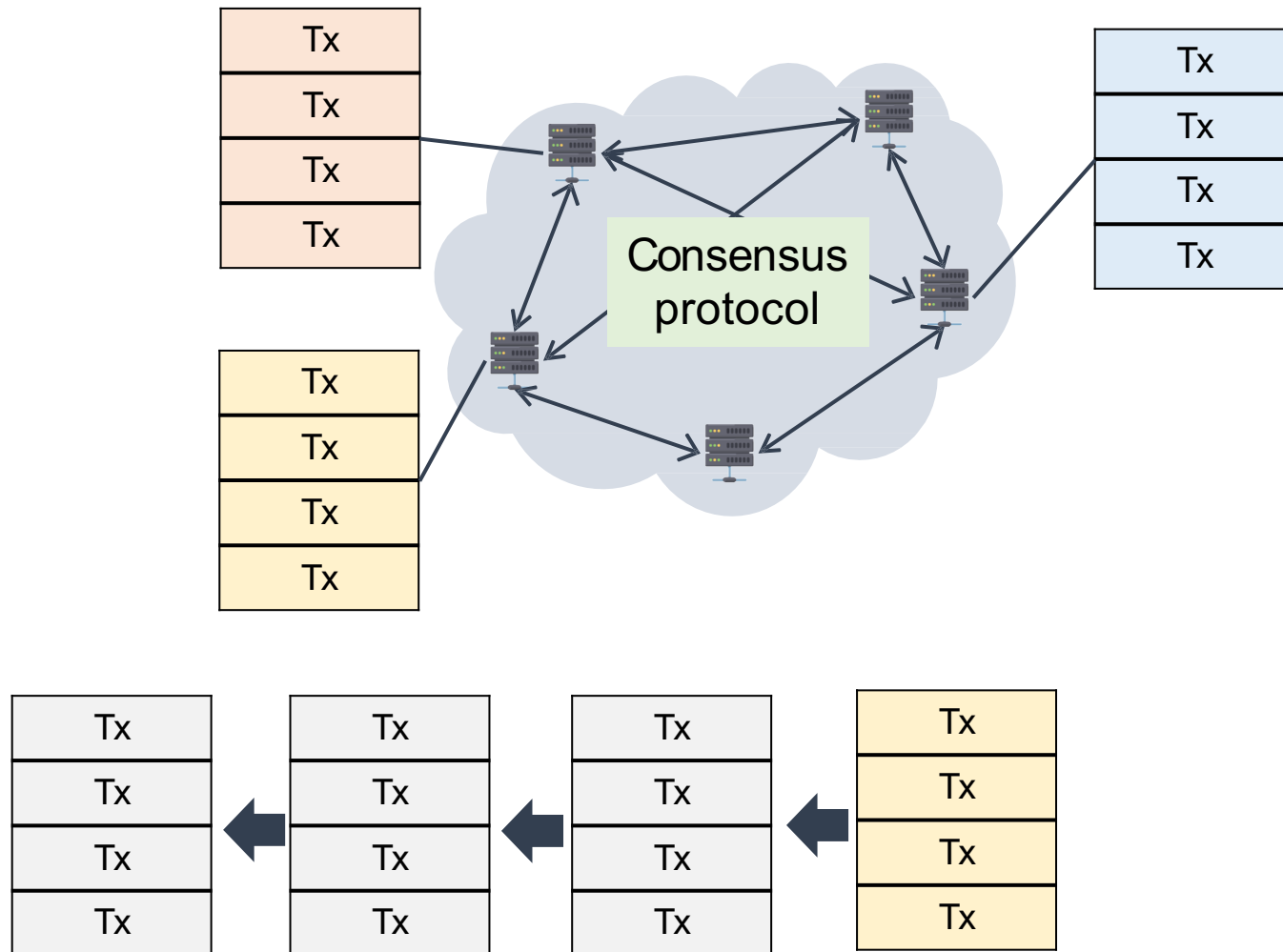
- To make a transaction, users **broadcast transactions** to the network nodes
- All nodes have a sequence of all blocks of **agreed transactions** they have reached consensus on
- Each node has a set of **outstanding transactions** (to be added to a block in the blockchain)

Distributed ledger via consensus



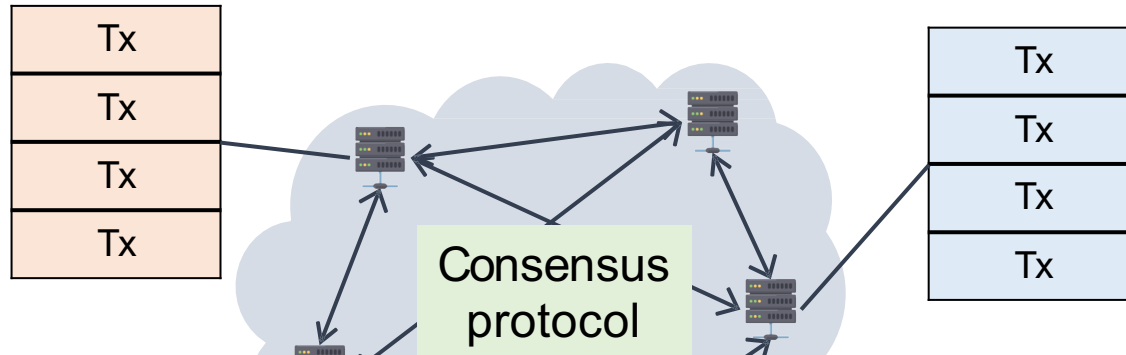
- To make a transaction, users **broadcast transactions** to the network nodes
- All nodes have a sequence of all blocks of **agreed transactions** they have reached consensus on
- Each node has a set of **outstanding transactions** (to be added to a block in the blockchain)

Blockchain: chain of blocks of transactions

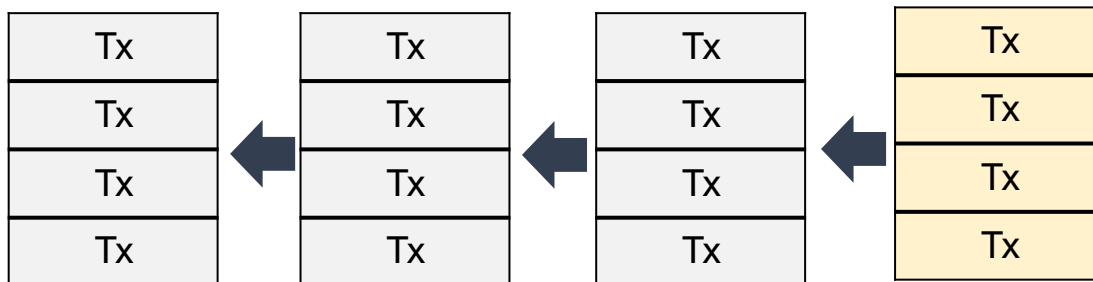
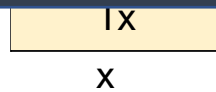


Chain of blocks (blocks point to previous blocks)

Blockchain: chain of blocks of transactions

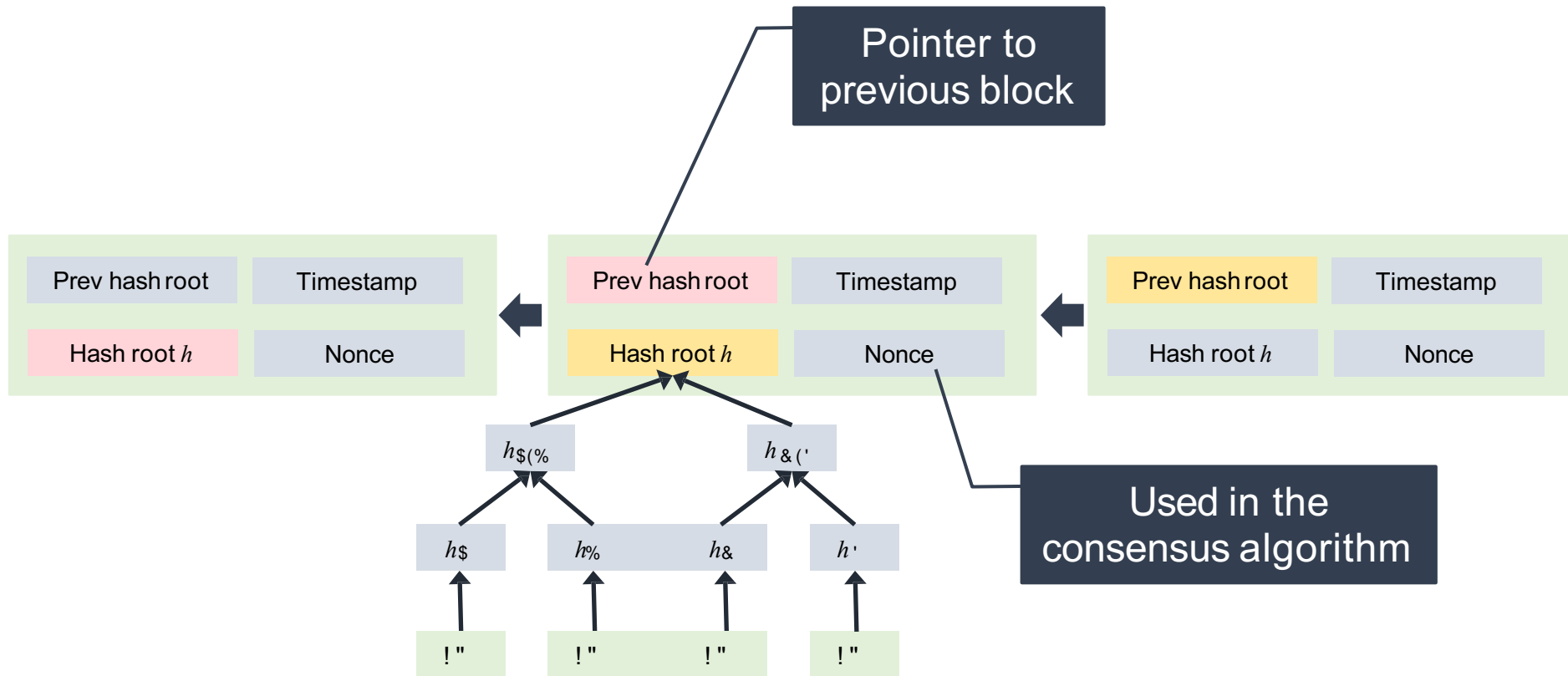


What is the structure of blocks in Bitcoin?



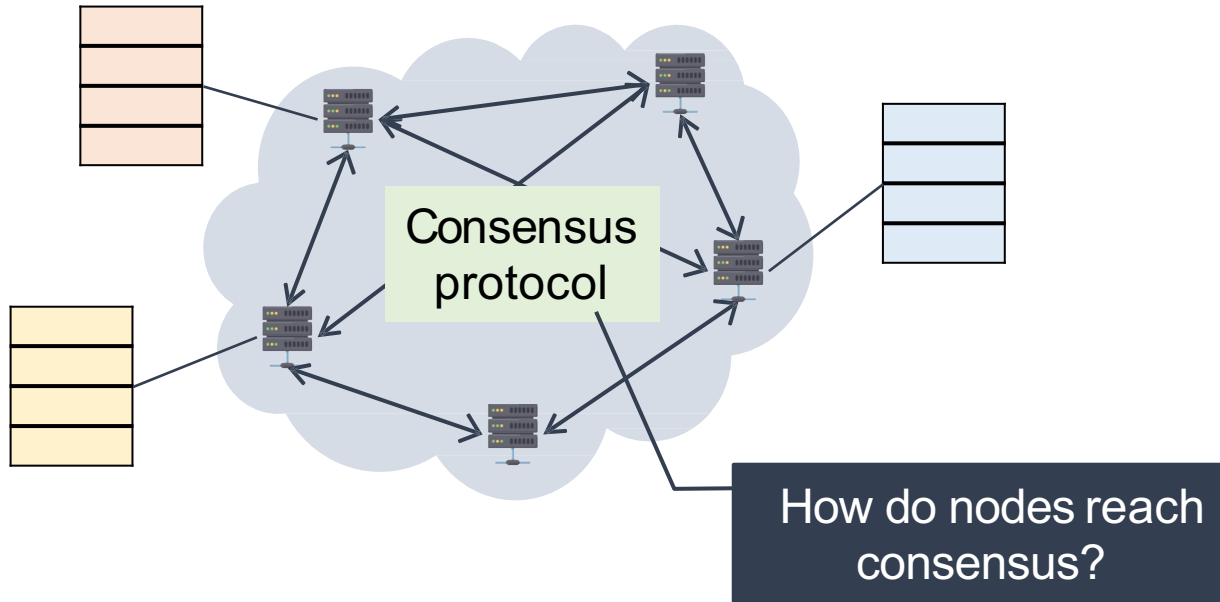
Chain of blocks (blocks point to previous blocks)

Blocks in Bitcoin



- Record integrity of all transaction in a merkle tree
- Enables efficient membership verification

What about consensus?



Consensus is hard

Nodes may **crash**

Nodes may be **malicious**

Network is **imperfect**

- Not all pair of nodes are connected
- Node failures
- No global time

Consensus algorithm (simplified)

1. New transactions are broadcast to all nodes
2. Each node collects new transactions into a block
3. In each round, a random node gets to broadcast its block
4. Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)
5. Nodes express their acceptance of the block by including its hash in the next block they create

How to select a random node?

Proof of work

To select a random node that extends the block:

- Select nodes in proportion to a resource that no one can monopolize

Several schemes

- In proportion to computing power: [proof-of-work](#)
- In proportion to ownership: [proof-of-stake](#)

Proof of work

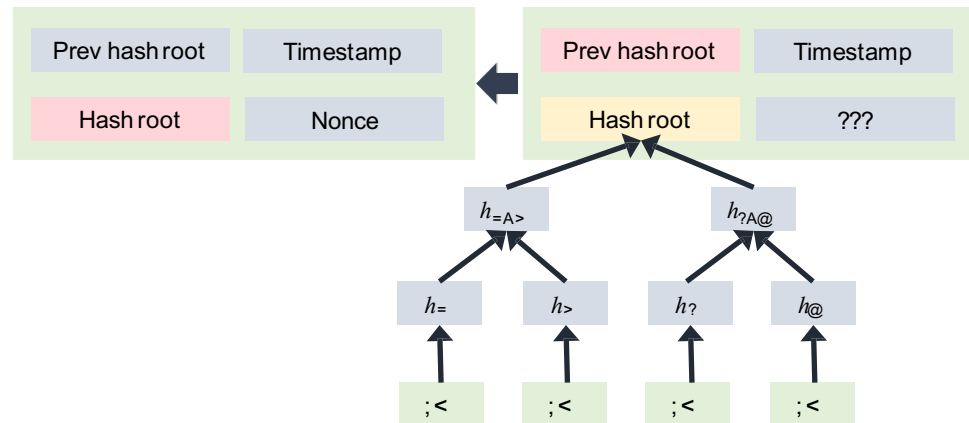
- Make nodes solve a hard puzzle

Output space of hash



Target space

Proof of work in Bitcoin



Given:

- Previous block with hash root h_{prev}
- Merkle tree consisting of all new transactions with root h_{tree}

Find:

- Nonce n such that

$$h(h_{\text{prev}} \parallel h_{\text{tree}} \parallel n) \leq \text{difficulty}$$

Properties of proof-of-work

Difficult to compute

- Current rate in Bitcoin: about 10 minutes per block
- Probability that a minor succeeds: $10 \text{ min} / \text{fraction of hash power}$

Parameterizable cost

- Nodes can adjust the target difficulty

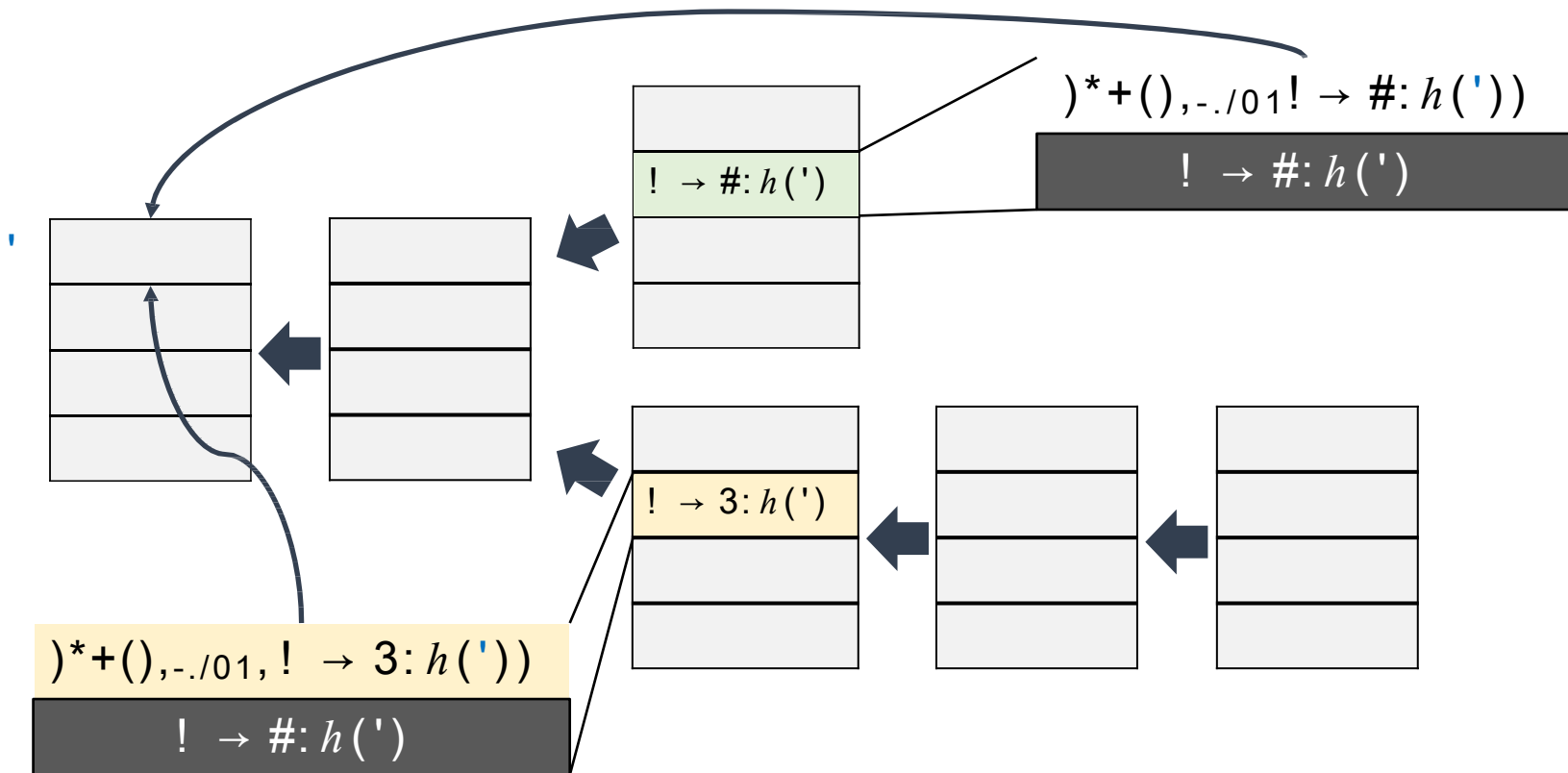
Easy to verify

- Other nodes verify that $H(h_{\text{parent}}, h_{\text{new}}) \leq \text{difficulty}$

Key security assumption

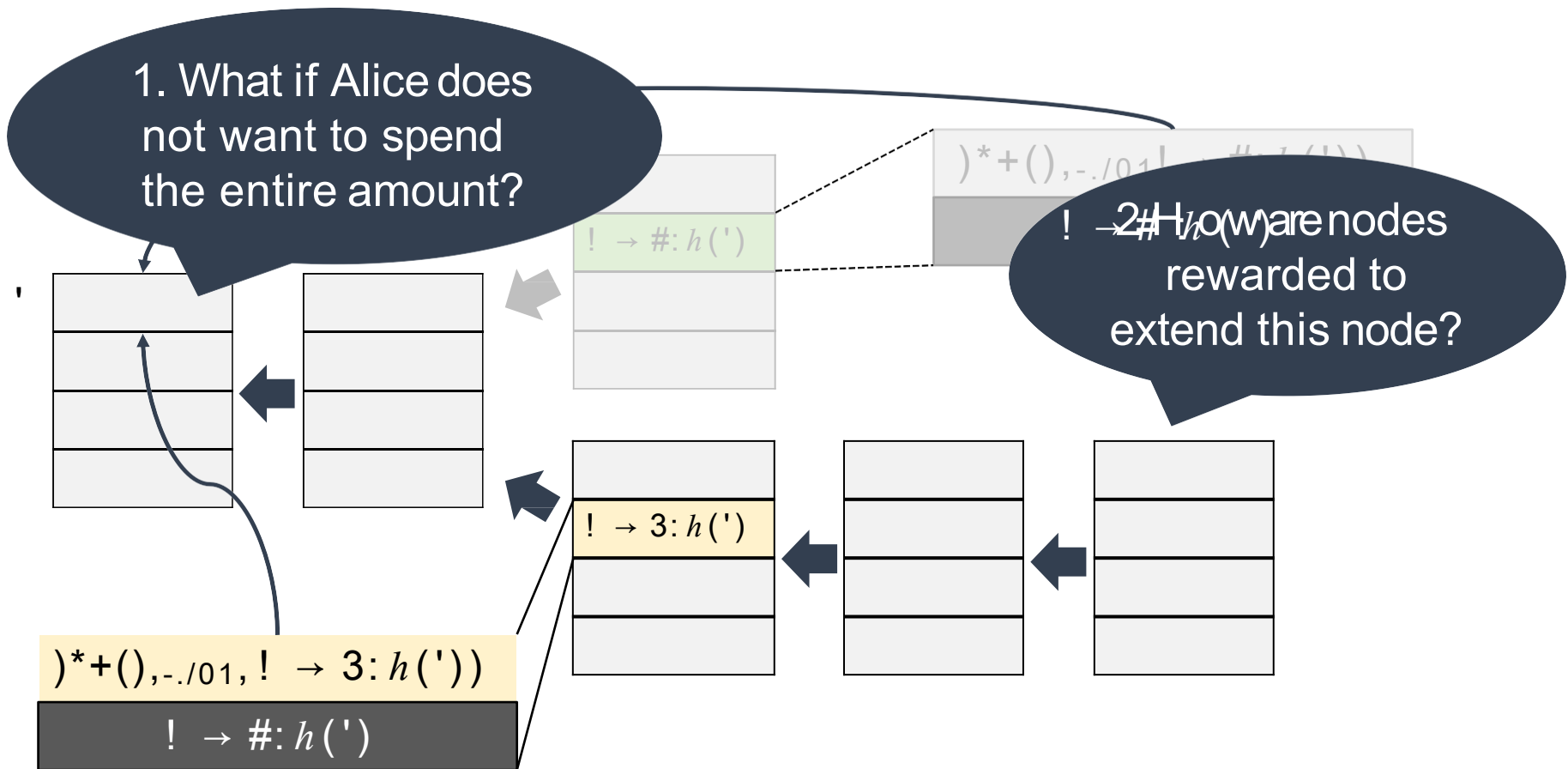
- Attacks infeasible if majority of miners weighted by hash power follow the protocol

Double spending



Honest nodes extend the longest valid branch

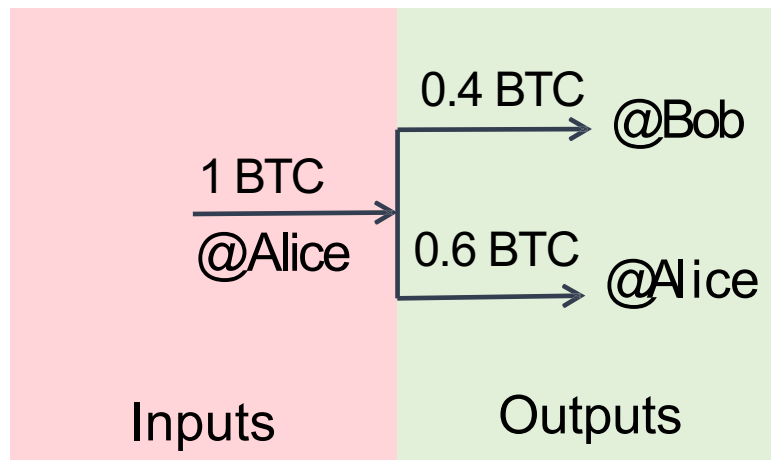
Double spending



Honest nodes extend the longest valid branch

Transactions

What if Alice does not want to spend the entire amount to Bob?



The transaction is valid if:

$$\sum \text{Inputs} \geq \sum \text{Outputs}$$

Transaction fee:

$$\sum \text{Inputs} - \sum \text{Outputs}$$

Incentivizing honest nodes to extend the longest chain

Block reward. The creator of a block can:

- Include special coin-creation transaction in the block
- Choose recipient address of this transaction
- Value of transaction is fixed
- Block creator "receives" the reward only if the block ends up on the long-term consensus branch

Transaction fees

- Creator of transaction may choose to make output value less than input value
- Remainder is transaction fee that goes to the block creator

- Dive into more details.

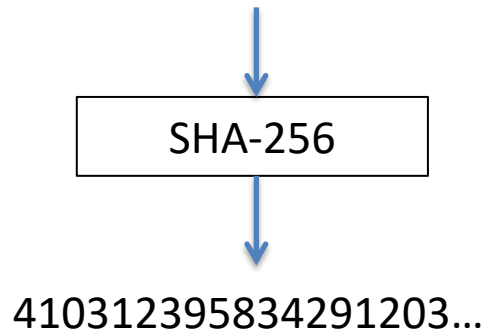
Welcome to Cryptoland

- Ugh. Do I *have* to learn all this detail?
- Yes. The laws of crypto are the laws of blockchains and bitcoin. Not understanding this will lead to bad intuitions about what this stuff can and cannot do.
- Luckily, only need to understand two laws of cryptography (and believe that people are motivated by incentives, I guess)
- We'll do this by building increasingly complex games that simulate parts of bitcoin and blockchains.

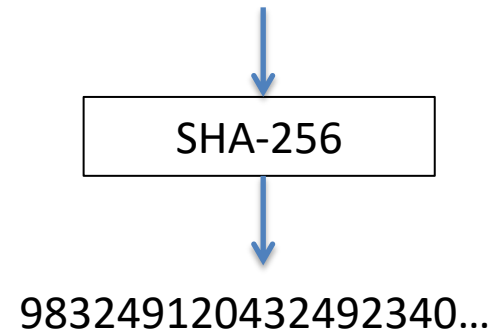
Ingredient #1: Hashes

- A hash function (like SHA-256) takes some data in, and produces an effectively random fixed size integer.
- Any change to the input randomizes it

“The quick brown fox did some crypto”



“The quick brown Fox did some crypto”



Hash-based Proof of Work

- Can't compute an input from an output
- To find a hash with N zeros at the start of the input, requires 2^N computations...proves computational work
- If we hash an incrementing “nonce” as the hash input, we can go looking for zeros:

in 3e-05 seconds, nonce = 0 yielded 0 zeros. value = 4c8f1205f49e70248939df9c7b704ace62c2245aba9e81641edf...

in 0.000138 seconds, nonce = 12 yielded 1 zeros. value = **0**5017256be77ad2985b36e75e486af325a620a9f29c54...

in 0.000482 seconds, nonce = 112 yielded 2 zeros. value = **00**ae7e0956382f55567d0ed9311cfd41dd2cf5f0a7137...

in 0.014505 seconds, nonce = 3728 yielded 3 zeros. value = **000**b5a6cfc0f076cd81ed3a60682063887cf055e47b...

in 0.595024 seconds, nonce = 181747 yielded 4 zeros. value = **0000**af058b74703b55e27437b89b1ebcc46f45ce55d6....

in 3.491151 seconds, nonce = 1037701 yielded 5 zeros. value = **00000**e55bd0d2027f3024c378e0cc511548c94fbeed0e....

in 32.006105 seconds, nonce = 9913520 yielded 6 zeros. value = **000000**77a77854ee39dc0dc996dea72dad8852afbde6...

in 590.89462 seconds, nonce = 186867248 yielded 7 zeros. value = **0000000**225060b16117b23dbea9ce6be86ac439d....

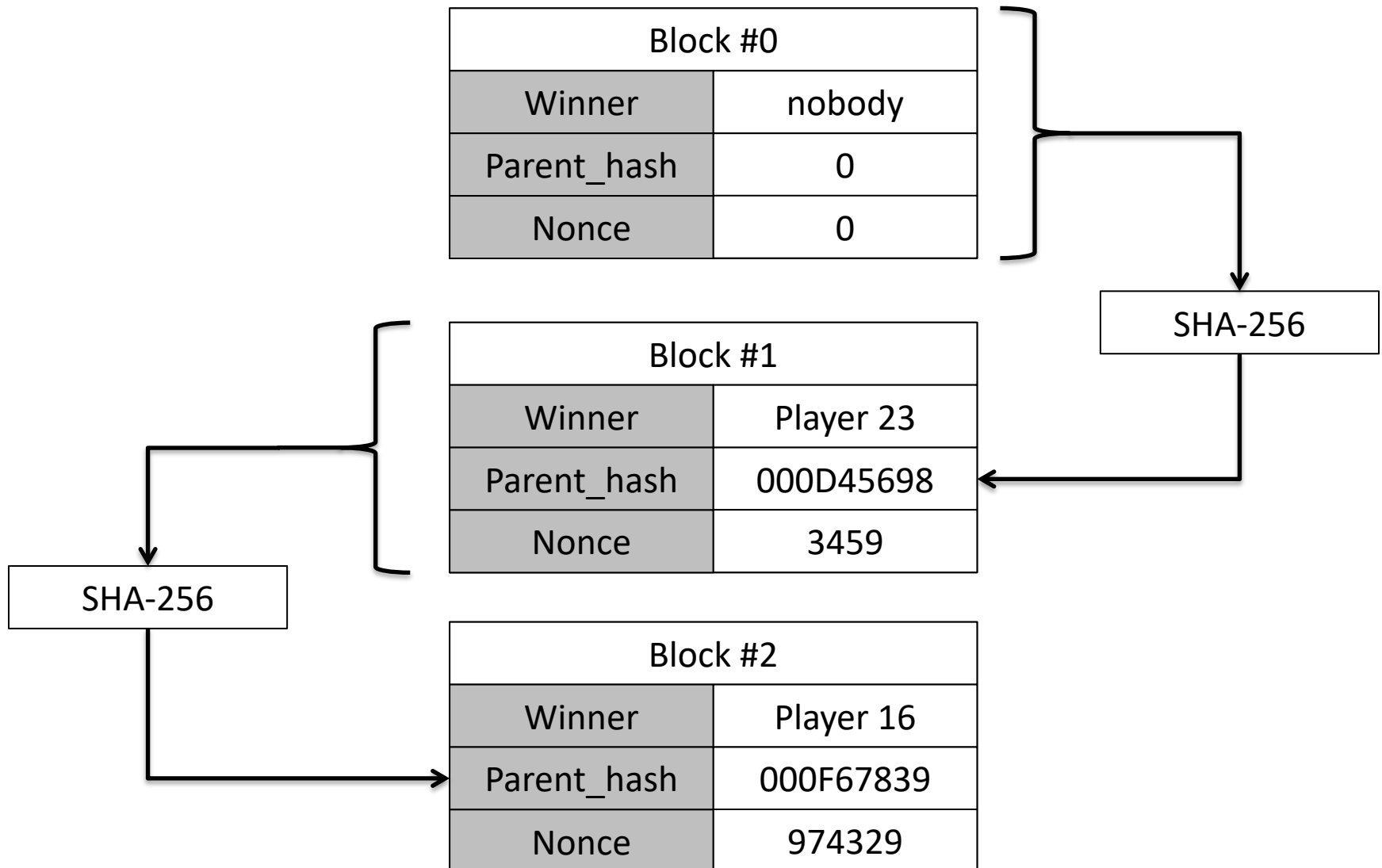
in 4686.171007 seconds, nonce = 1424462909 yielded 8 zeros. value = **00000000**2dd743724609a9f57260e2492908d....

We can now make this into a distributed “game”

Game #1 – The Chain Race

- A parameter N sets the difficulty of the game
- Players get a list of blocks, with:
 - A block number
 - A winner number
 - A nonce value
 - A hash of the previous block
 - A hash of the current block with N zeros
- Players accumulate points by creating blocks
 - Hash the previous block
 - Find a hash of the new block with enough zeros
 - They then transmit this block to everyone

Game #1 – The Chain Race



The Nonce / Hash Loop

- The algorithm to make a new block:
 1. Verify the hashes of all the previous blocks
 2. Build a new block with a random nonce
 3. Hash the new block. Does it have N zeros?
 - No? Go back to Step 2
 - Yes? Send your new block to everyone!
- Note that as a result of step #1, you can find out how many points anyone has by counting how many blocks they have won

How hard is the game?

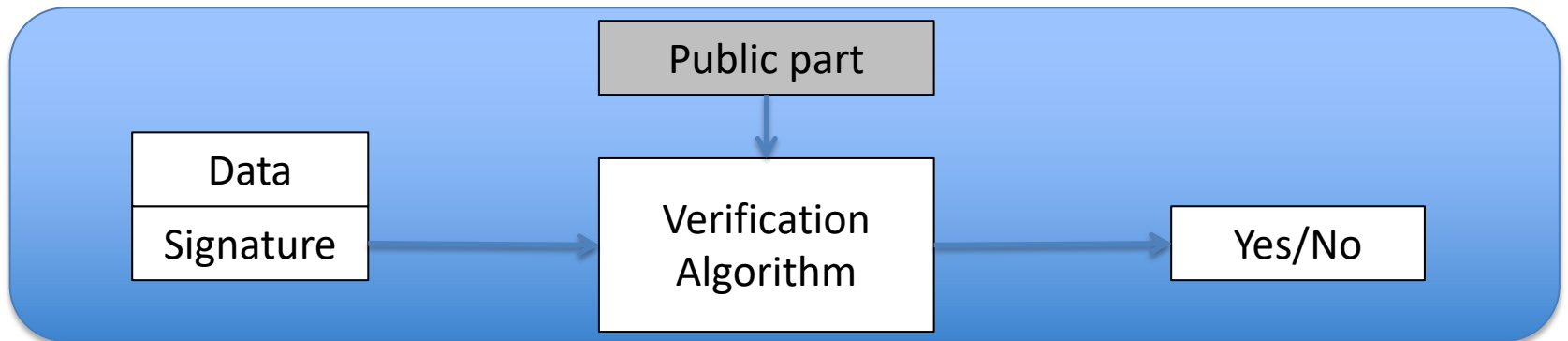
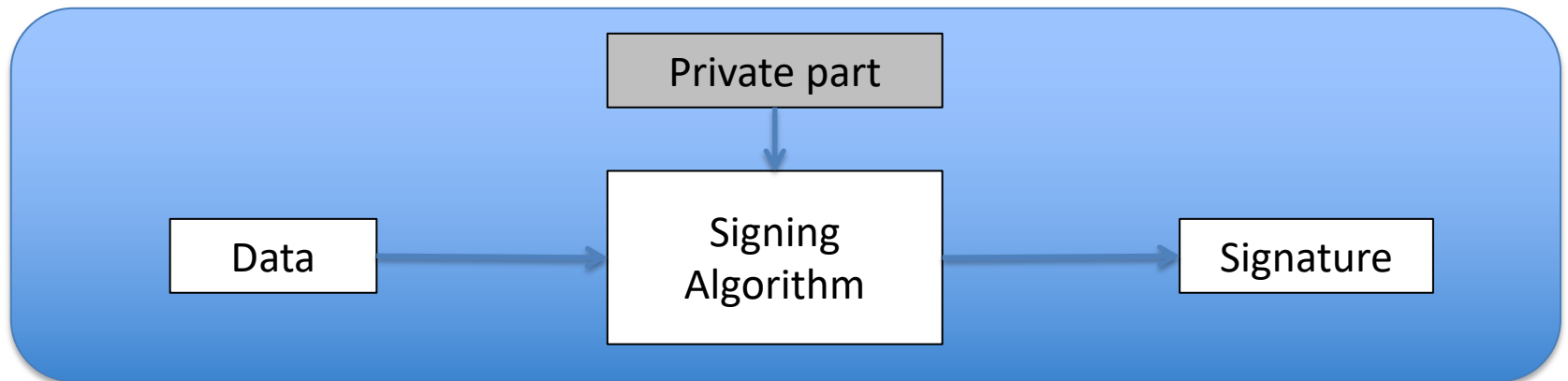
- For N zeros, because the SHA-256 output is effectively random, getting zero bits = same as flipping a coin and getting N heads in a row
- For N zeros, have to try $2^N/2$ nonces...
 - N=1 Try 1 nonce
 - N = 16 ... Try 32768 nonces
 - N = 32 ... Try 2 billion nonces
- Winning a block proves the player did work

What about cheaters?

- One way to cheat: make up a fake hash!
- What happens then?
 - Step 1 in the algorithm will fail for all the other players.
 - Other players will not use your block, making it not part of the chain

Ingredient #2: Digital Signatures

Signing key	
Public part	454F4D3E1..
Private part	56F23F2D..



Trading points

Make player ID = public key

We can now make trades by signing messages and sending them to everyone

Signed trades are:

- Unalterable
- Verifiable by anyone
- From key to key, not tied to a “real” identity

Trade #8423	
From	Public_key1
To	Public_key2
Amount	50 points
Signature	345349354

Trade #8424	
From	Public_key2
To	Public_key3
Amount	50 points
Signature	734589345

Game #2 – The Race with Trades

Block #0	
Winner Key	nobody
Parent_hash	0
Nonce	0



Block #1	
Winner Key	045F45F...
Parent_hash	000D45698
Nonce	3459



Block #2	
Winner Key	8234DB4...
Parent_hash	000F67839
Nonce	3459

Trade #8423	
From	Public_key1
To	Public_key2
Amount	50 points
Signature	345349354

Trade #8424	
From	Public_key2
To	Public_key3
Amount	50 points
Signature	734589345

Cheating!

- Can't alter transactions, but sneaky players could trade extra points by sending more trades than they have points to cover
- “Overtrading” not resolvable, because don't have an absolute unalterable source of time
- Let's fix this in game #3...
 - Critical insight: Put the trades in the blocks.

Game #3 – No-cheating Social

Block #2													
Winner_key	6B34C03...												
Parent_hash	004539A3F												
Nonce	54695												
<table><tr><th colspan="2">Trade #5</th></tr><tr><td>From</td><td>Public_key1</td></tr><tr><td>To</td><td>Public_key2</td></tr><tr><td>Amount</td><td>50 points</td></tr><tr><td>Signature</td><td>345349354</td></tr></table> <table><tr><th colspan="2">Trade #6</th></tr></table> ...		Trade #5		From	Public_key1	To	Public_key2	Amount	50 points	Signature	345349354	Trade #6	
Trade #5													
From	Public_key1												
To	Public_key2												
Amount	50 points												
Signature	345349354												
Trade #6													

Game #3 is magic...

- Players expend effort to get points
- Players can trade points securely
 - Signatures prevent alteration of trades
 - Signatures authenticate the origin of trades
- Players can detect overtrading
 - Players will decline to extend the game on blocks with overtrades
 - If they do, they are wasting effort, since other players will not extend the game on their blocks

Game #3 Problems

- Why bother to put trades in your block?
- Lets solve this by adding a fee in transactions
 - Incent players to add transactions by giving them points per trade added
 - Two ways to get points!
- Why limit trades to players?
 - Let players send points to anyone with a public key....
 - This is now a global transaction system

Game #4 – Simplified Bitcoin

- Players = “miners”, points = “bitcoins”
- Transactions send value (bitcoins) from key to key
- The chain race game (blockchain) prevents overspending without a central authority
- Game rules = bitcoin node code, changes by miner consensus
- Player consensus replaces authority
 - Number of coins (limit to 21 million)
 - Reward per block
 - How difficulty grows

Transition to transactions

- Note that player/miners can interact with non-players
- Once a point is created, the recipient can create a transaction to any public key
- Now can extend to trades with non-miner/players
- All points still originate with some block/miner