CS 4770: Cryptography

CS 6750: Cryptography and Communication Security

Alina Oprea
Associate Professor, CCIS
Northeastern University

March 30 2017

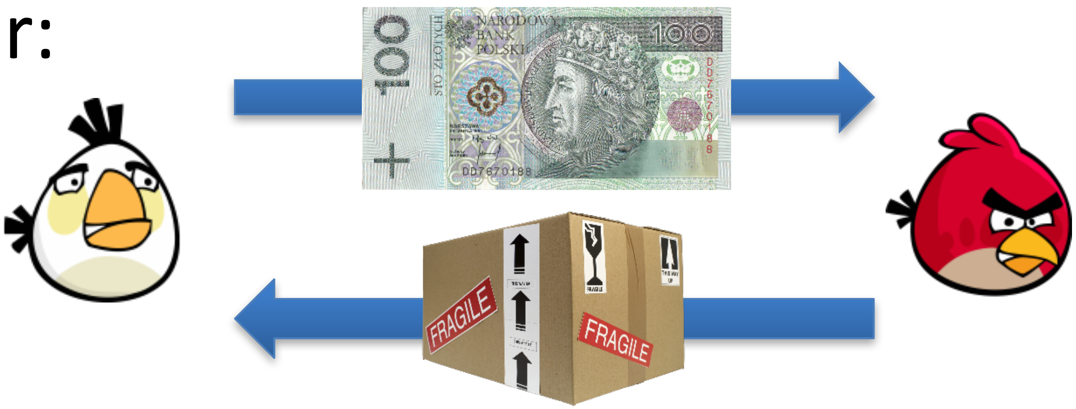
Outline

- Digital currencies
  - Advantages over paper cash
- Bitcoin design goals
  - Decentralized
  - Publicly verifiable
  - Pseudo-anonymity
- Design principles
  - Based on computationally hard cryptographic puzzles
  - Assumes honest majority
  - Economic incentives to players to be honest

Digital currencies

Digital vs. paper currencies

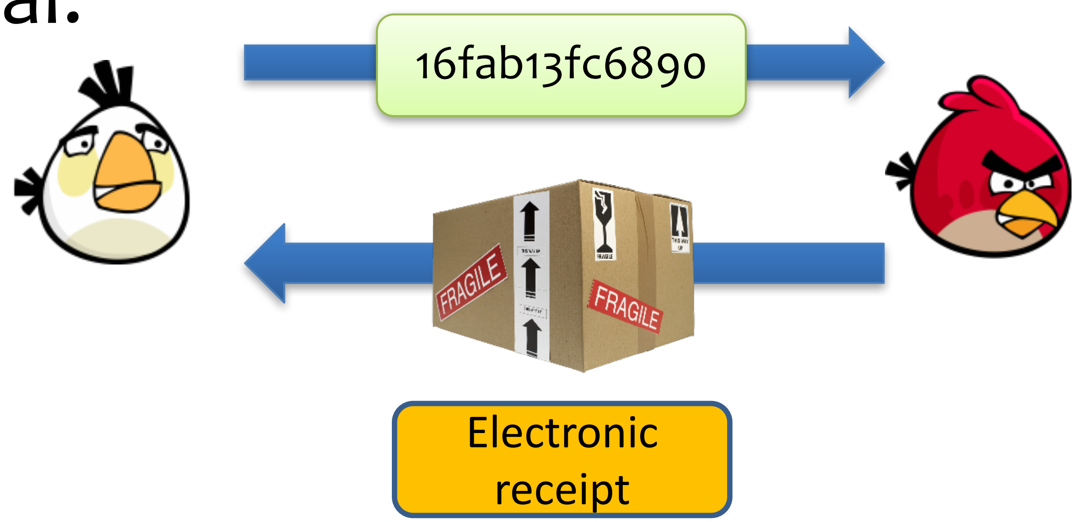
Paper:



Advantages of paper

- Portable
- Cannot double-spend
- Non-repudiable
- Semi-anonymous (have serial numbers)

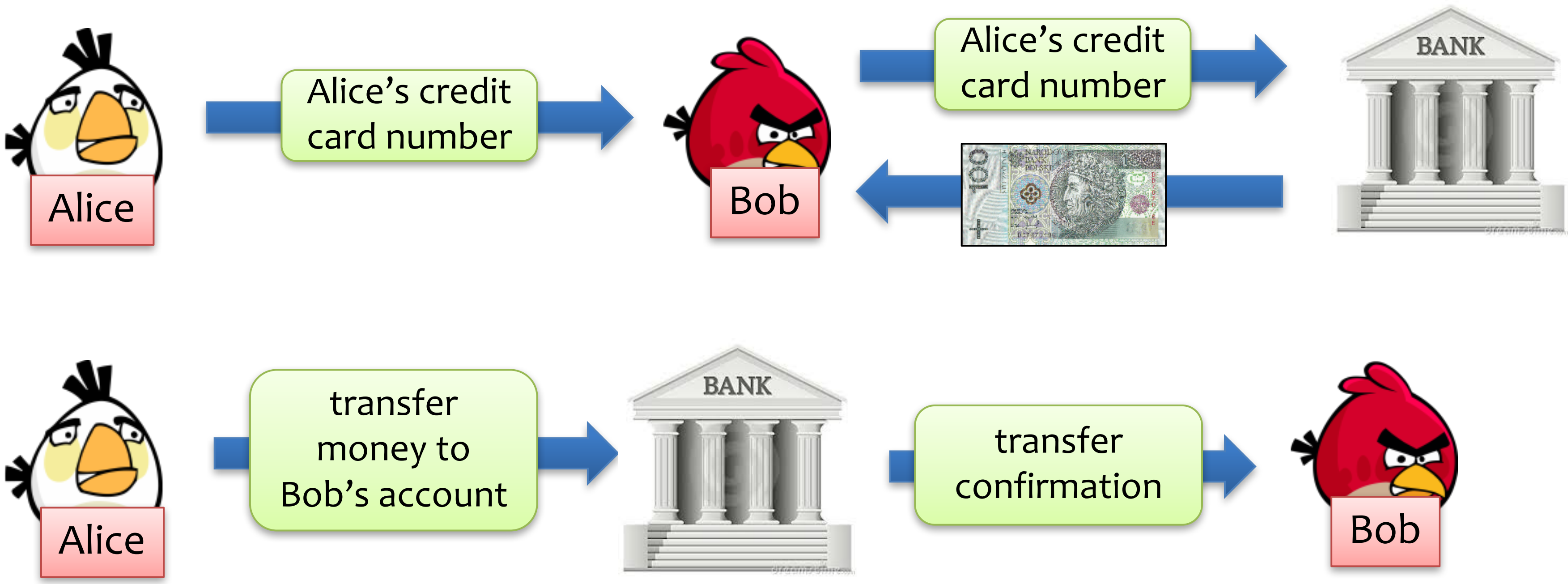
Digital:



Disadvantages of paper

- Easy to steal
- No tax record
- Trust in central authority
- Doesn’t work online

Traditional ways of paying "digitally"



BENEFITS

1. Convenient (pay online)
2. Highly regulated
3. Banks handle fraud
4. Cannot double-spend
5. Tax records

PROBLEMS

1. **Trusted server** for each transaction
2. High **transaction fees**
3. Record of all transactions (**No anonymity/privacy)**

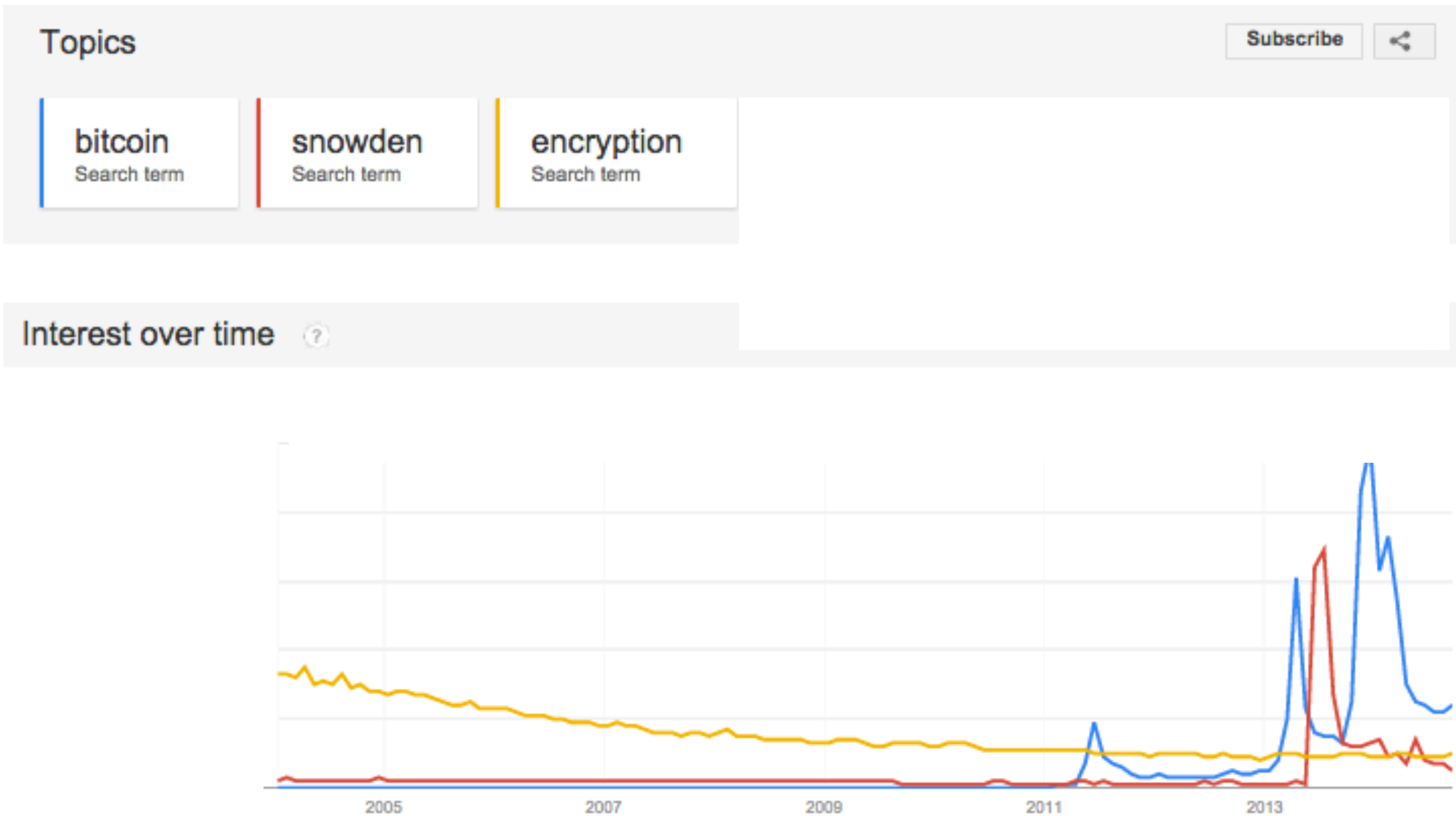
Bitcoin – a "digital analogue" of the paper money

A digital currency introduced by "Satoshi Nakamoto" in 2008

- First e-cash without a centralized issuing authority
  - Store and transfer value without reliance on central banks
  - Anyone can join the system and make transactions
  - Transactions are publicly verifiable
- Built on top of an unstructured P2P system
  - Participants validate transactions and mint currency
  - System works as long as the *majority of users are honest*
  - Provides economic incentives for users to be honest

Currency unit: **Bitcoin (BTC) 1 BTC = 10^8 Satoshi;** value ≈ $1250

Probably one of the most discussed cryptographic technologies ever!



Bitcoin

in Bitcoin:

No trusted server, money circulates

Low fees

“Pseudonymity”

PROBLEMS WITH DIGITAL PAYMENT

1. **Trusted server** for each transaction
2. High **transaction fees**
3. **No anonymity/privacy.**

Bitcoin ≈ “real money”?

Bitcoin value comes from the fact that:

“people expect that other people will accept it in the future.”

enthusiasts:

It’s like all the other currencies

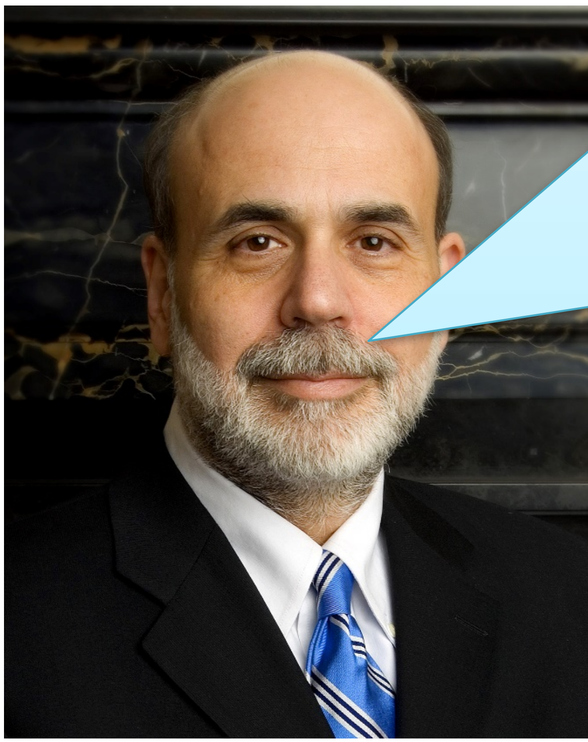
sceptics:

P. Krugman

A. Greenspan

It’s a Ponzi scheme

Some economists are more positive

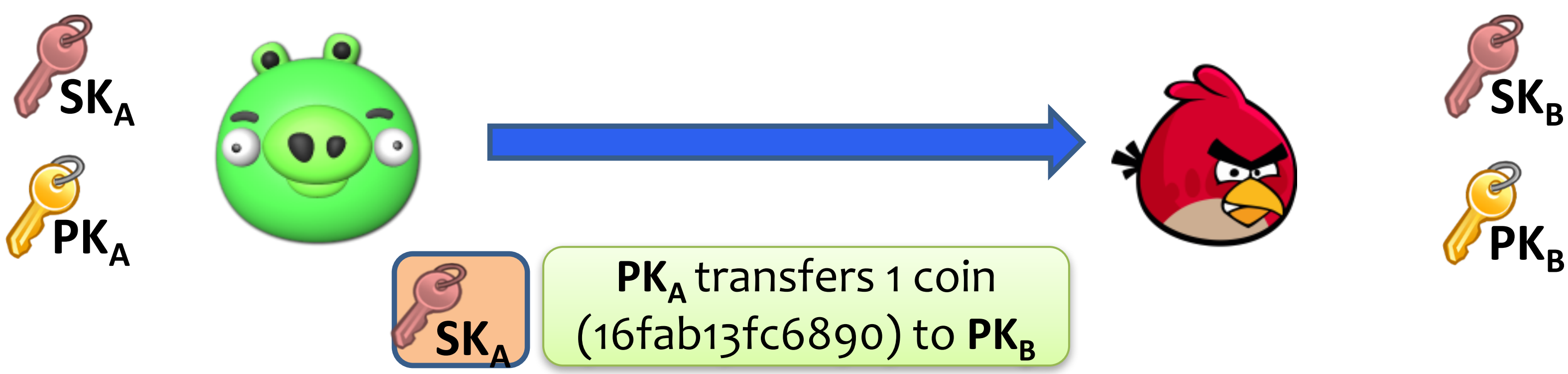


Ben Bernanke

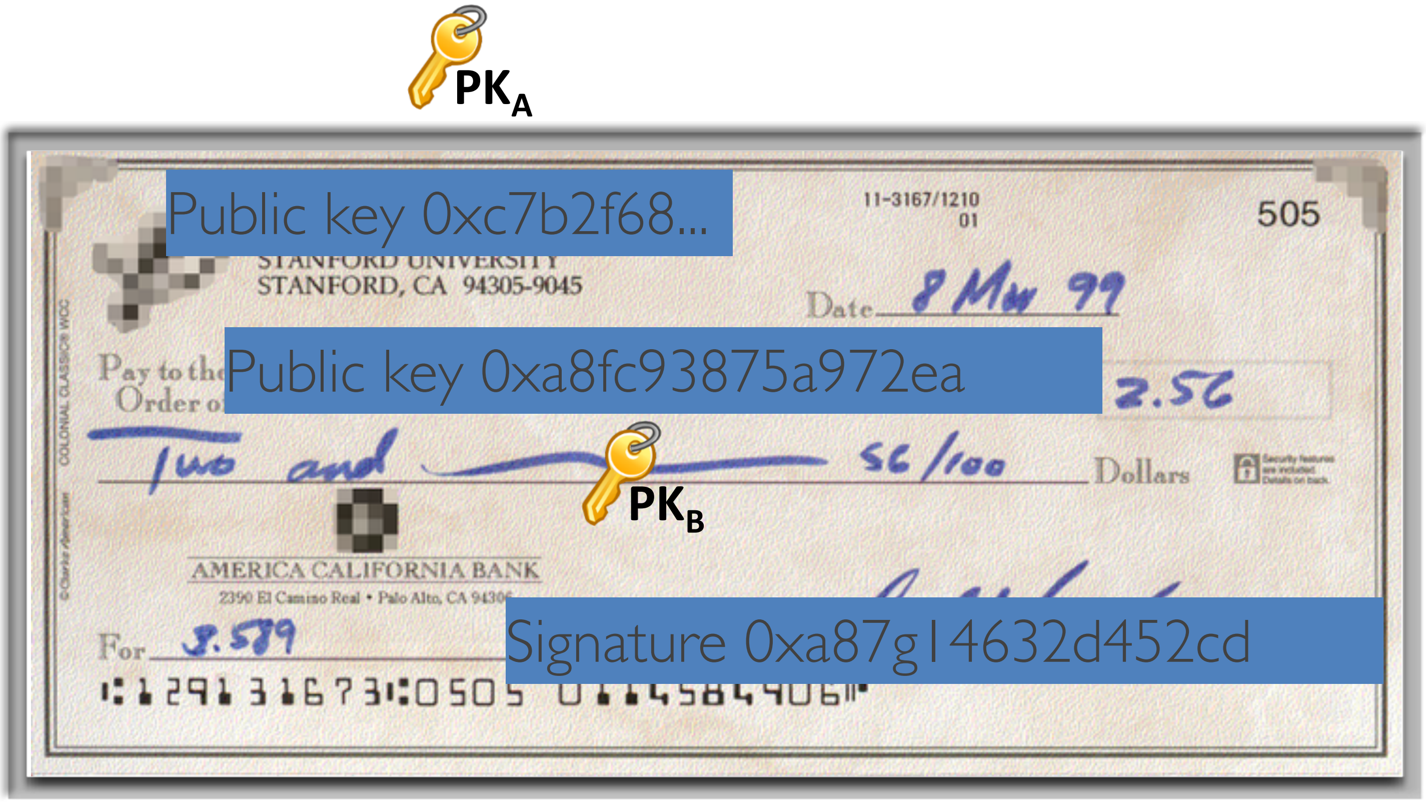
While these types of innovations **may pose risks** related to law enforcement and supervisory matters, there are also areas in which they may hold long-term promise, particularly if the innovations **promote a faster, more secure and more efficient payment system**.

Strawman protocol

- Alice owns a coin and wants to transfer to Bob
  - Transactions can not be forged
  - Non-repudiable (can not be reversed)
  - Spend once every coin
  - Can be spent by Bob later
- Format of coin?
  - Unique serial number (long bit string)
- What to use for identities?
  - Requirement for weak identities (no use of national ID or passport)
  - Public keys!



Bitcoin Transactions



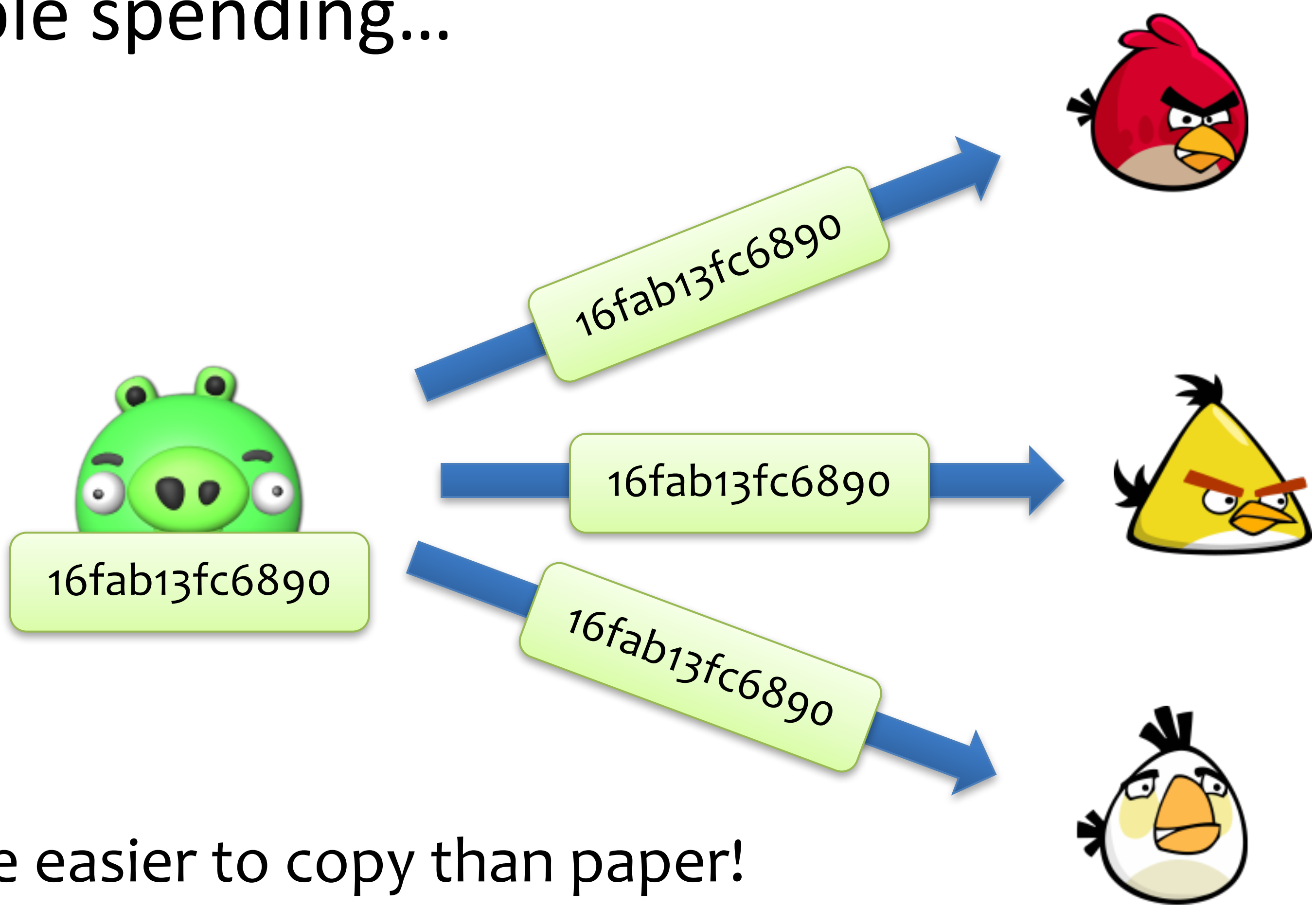
Serial number

Value

Digital signature

Main problem with the digital money

Double spending…



Bits are easier to copy than paper!
Signatures alone do not prevent this

Bitcoin idea

Public trusted bulletin-board (public ledger)

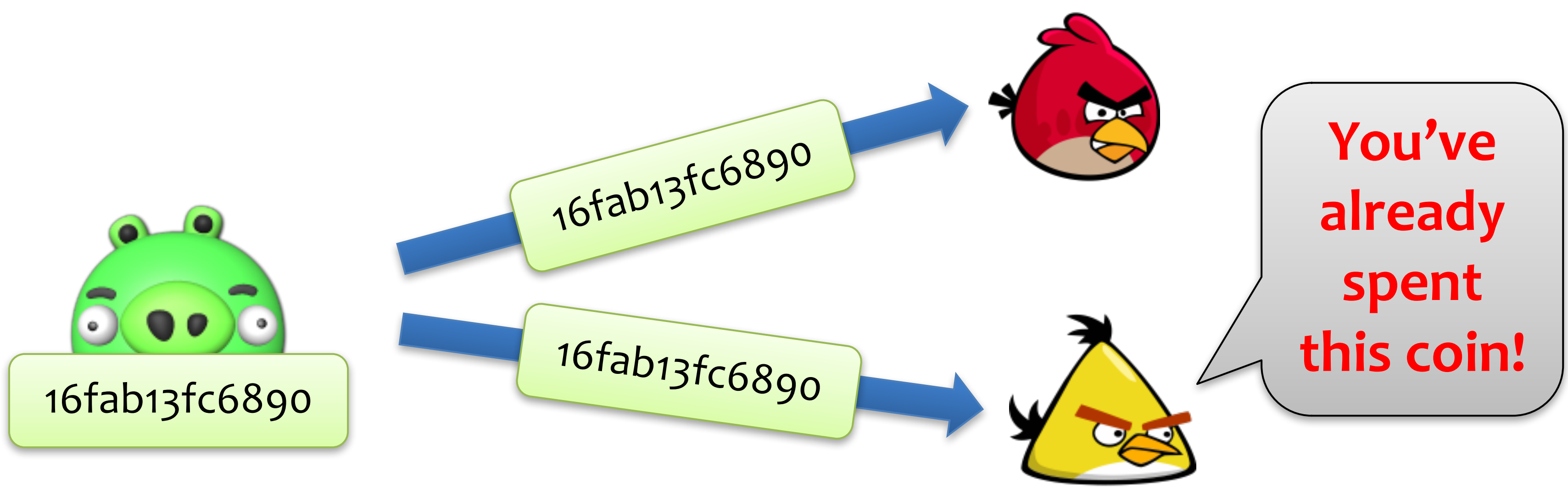
- Includes list of all transactions
- Verifiable by all users
- Decentralized
- Maintained jointly by all users

Public ledger

PK_A owns coin 92cab79341c

SK_A — PK_A transfers 1 coin (16fab13fc6890) to PK_B

SK_A — PK_A transfers 1 coin (16fab13fc6890) to PK_C

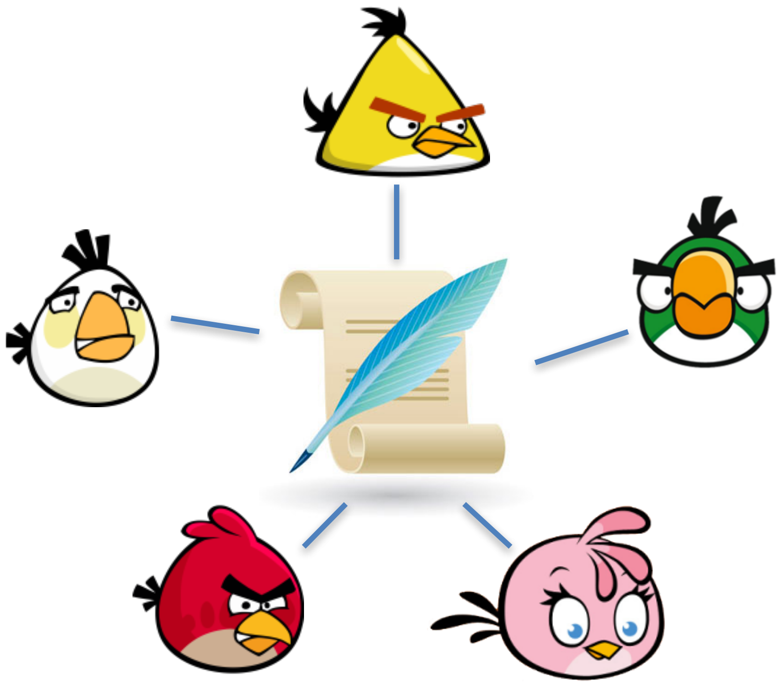


What needs to be discussed

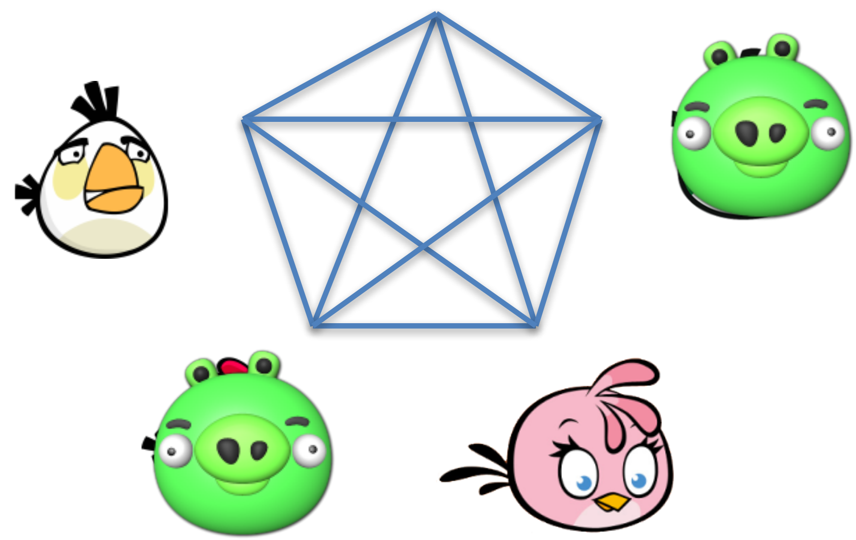
1. How is the **trusted bulletin-board** maintained in decentralized manner?
2. How to prevent attackers to obtain majority ?
3. How are users incentivized to be honest?
4. What is the syntax of the transactions?

Trusted bulletin-board emulation

the "ideal" world



a **protocol** that emulates the ideal world



Main difficulty: Some parties can cheat.

Classical result: Consensus is possible if the "majority is honest".

For example for **5** players we can tolerate at most **2** "cheaters".

Consensus

- Goals
  - Multiple players agree on same value
  - The protocol terminates and all correct nodes decide on the same value
  - This value must have been proposed by some correct node
- Consensus in Bitcoin
  - Nodes agree on valid transactions and their order
  - Broadcast model: nodes broadcast messages to all other nodes
  - Assume majority of correct nodes
- Challenges
  - Nodes might crash or be outright malicious
  - Network is imperfect (not all nodes are online)
  - Highly distributed, variable latency
    - Implications: there is no notion of global time; transactions can not simply be ordered by timestamps
  - Impossibility results for general consensus problem in completely asynchronous model

Key insights

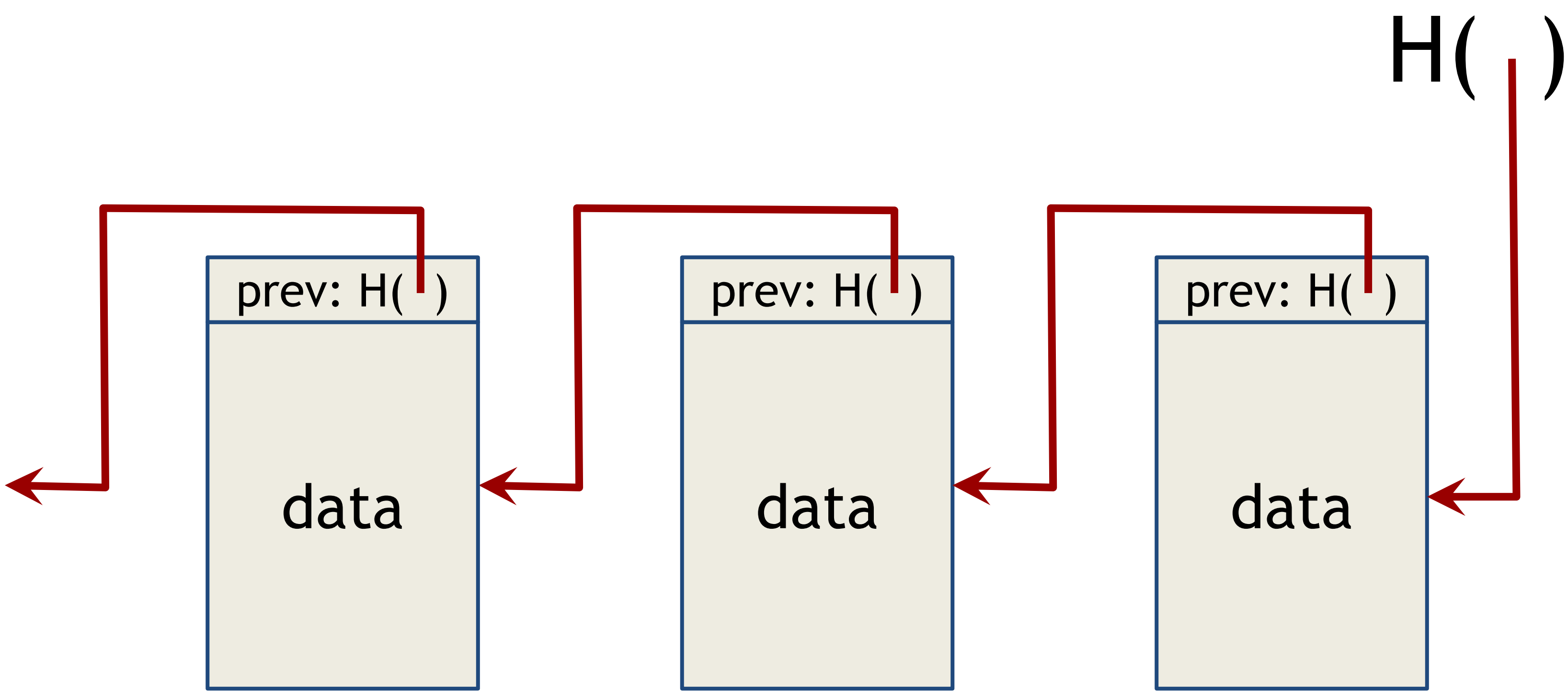
- Bitcoin is a P2P network of nodes (miners)
  - Each transaction is broadcast to all nodes
  - Each node keeps a log (ledger) of **all** Bitcoin transactions
  - New transactions are verified and appended to log
- Tamper-evident log
  - Valid transactions can not be modified
- Consensus happens over longer periods of time
  - Probabilistic guarantees
  - In online transactions can have some delay
- Provide financial incentives for nodes to be honest
  - Bitcoin does not solve the general consensus problem, but achieves consensus for digital currencies

Tamper-evident log

- Bitcoin is a P2P network of nodes (miners)
  - Each transaction is broadcast to all nodes
  - Each node keeps a log (ledger) of **all** Bitcoin transactions
  - New transactions are verified and appended to log
- Tamper-evident log
  - Valid transactions can not be modified
  - How to design it?

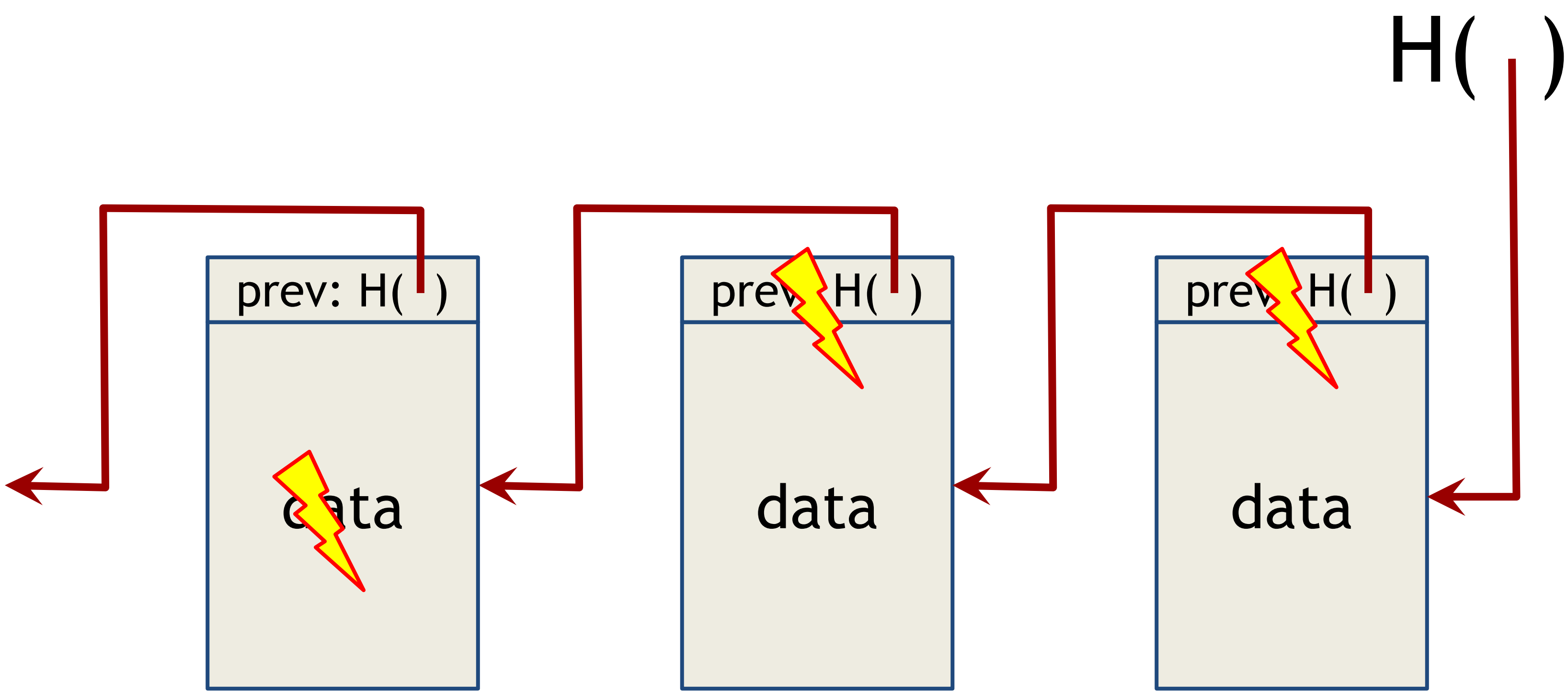
Block chain

Linked list with hash pointers



Tamper-evident log

Detecting tampering

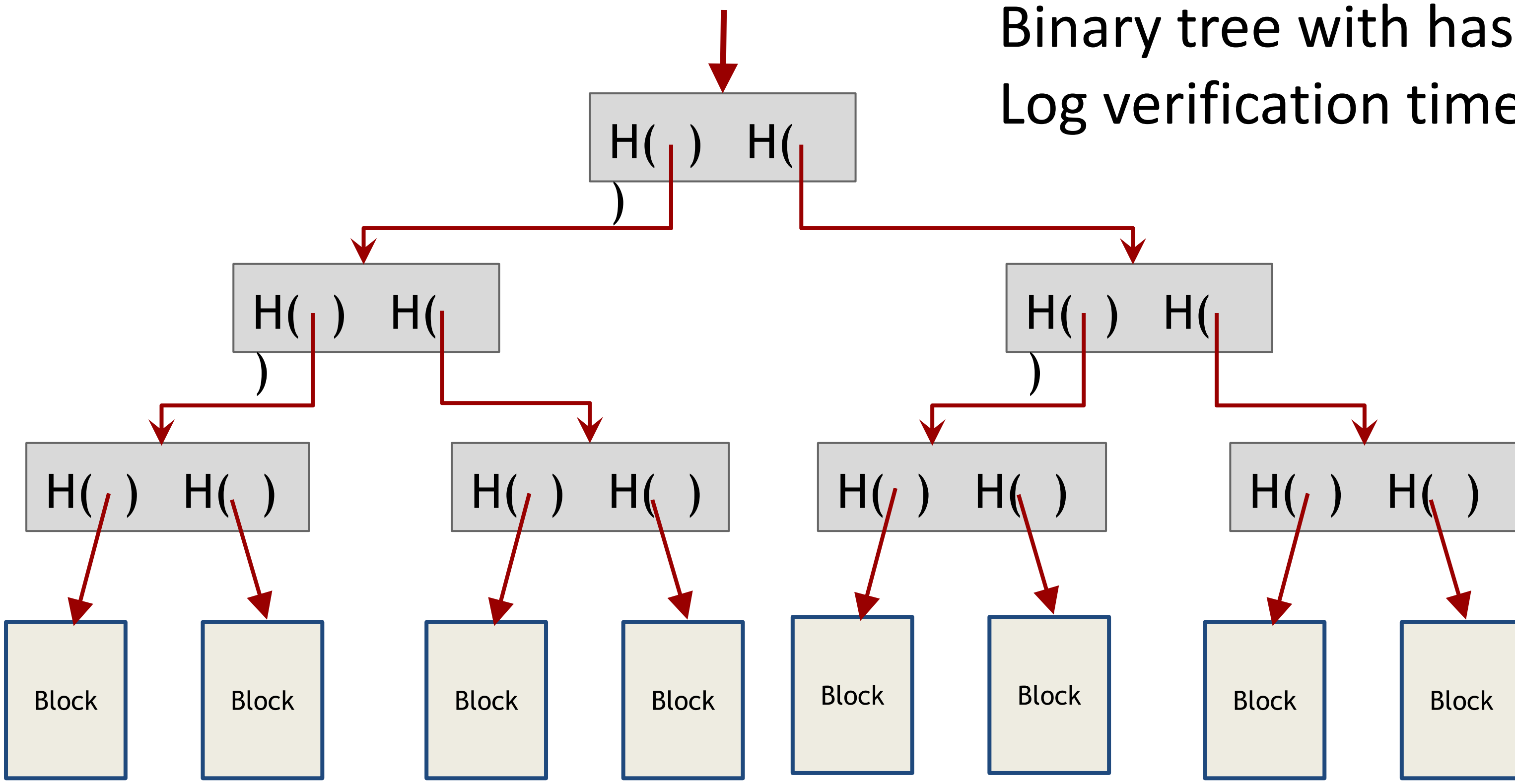


Tamper-evident log

Merkle trees

Binary tree with hash pointers

Log verification time



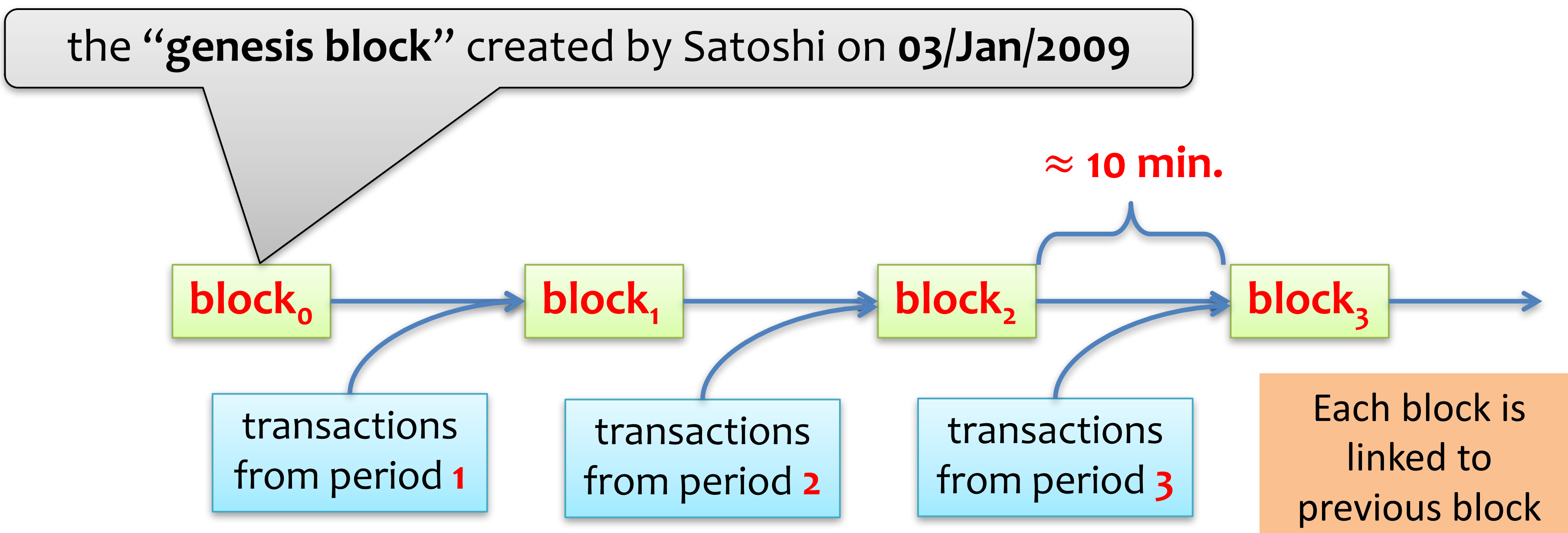
Block

Block Hash	Prev. Block Hash	Nonce
Transaction *I*		
Transaction *J*		
Transaction *K*		

Block chain

The users participating in the scheme are called “miners”.

They maintain a chain of blocks (blockchain):



What needs to be discussed

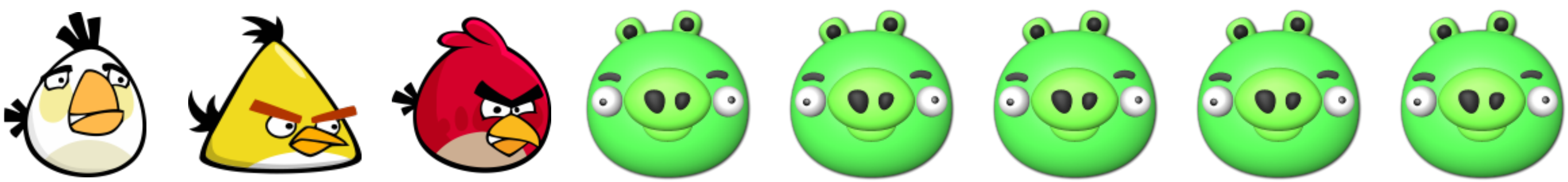
1. How is the **trusted bulletin-board** maintained in decentralized manner?
2. How to prevent attackers to obtain

majority ?

1. How are users incentivized to be honest?
2. What is the syntax of the transactions?

Problem

How to define "**majority**" in
a situation where
everybody can join the network?



Sybil attacks – users create multiple identities
Attacker can control majority!

The Bitcoin solution

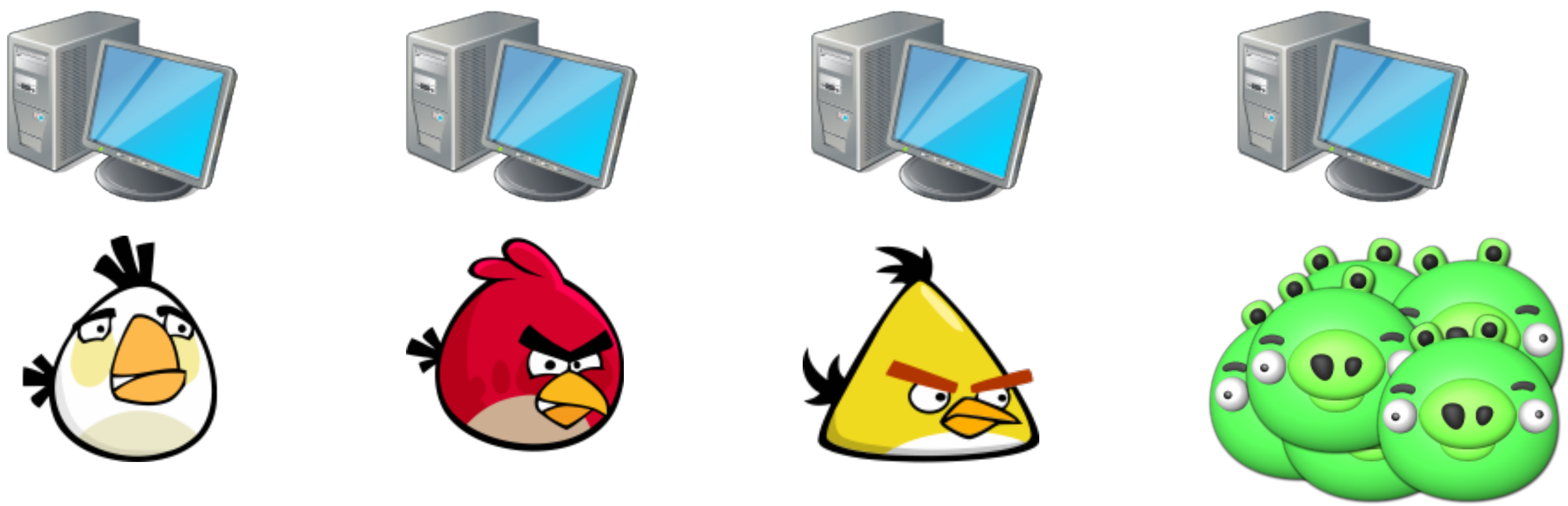
Use a resource that is hard to obtain

- In the past gold, could use national/state IDs (do not have anonimity)

Key insight: use computational resource (CPU power)

- Users need to present Proofs-of-Work to append transactions to ledger

Now creating multiple identities does not help!



Proofs of Work (PoW)

Properties

- Cryptographic puzzles users need to solve
- Take minimum amount of CPU resources to compute
- Fast to verify

- Incentivize honest users to constantly participate in the process
  - The honest users can use their **idle CPU cycles**
  - **Nowadays**: often done on **dedicated hardware**
- Alleviates Sybil attacks
  - E.g. one machine pretending to be 100 Sybils doesn't magically get 100x CPU power
  - Attackers need to consume 100x computational resources
  - Implicit assumption: no single entity can control the majority of computational power

A simple hash-based PoW

H -- a hash function whose computation takes time **TIME(H)**

random **x**

Prover
finds **s** such that
H(s,x) starts with **n** zeros (in binary)

salt	hardness parameter n

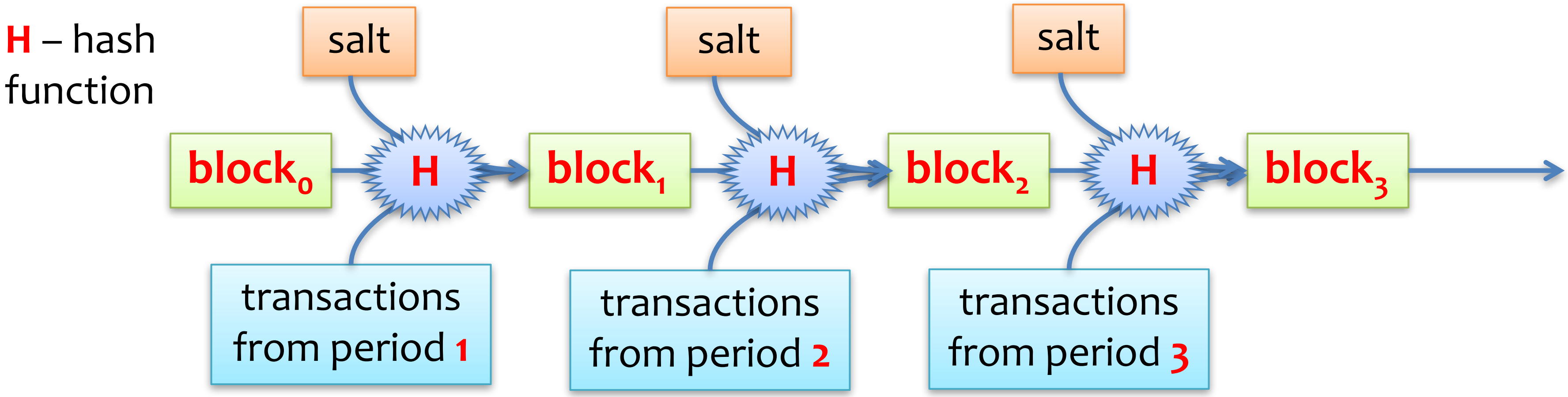
takes time $2^n \cdot$ **TIME(H)**

s

Verifier
checks if
H(s,x) starts with **n** zeros

takes time **TIME(H)**

How are the PoWs used?



<u>Main idea</u>: to extend it one needs to find **salt** such that

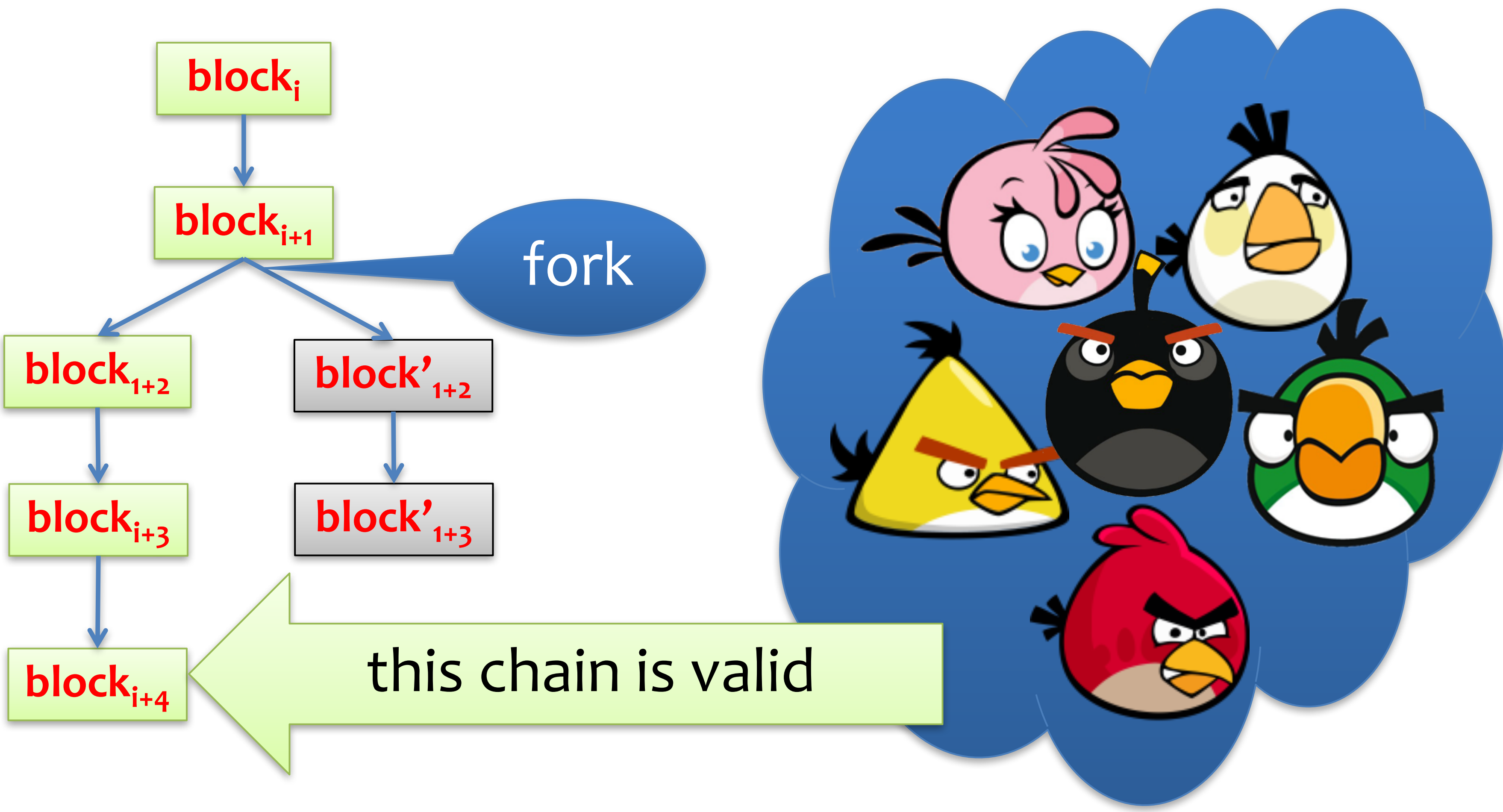
H(salt, block$_i$, transactions) starts with some number **n** of **zeros**

Process is called block mining

What if there is a “fork”?

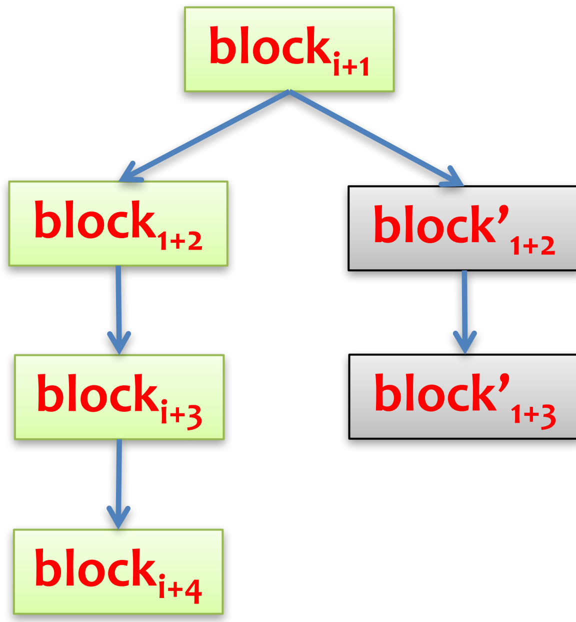
The “**longest**” chain counts.

- It includes “more work”



Does it make sense to “work” on a shorter chain?

No!



Because everybody else is working on extending the longest chain.

Recall: we assumed that the majority follows the protocol.

Dropped blocks

- Reasons for which valid blocks are not eventually included in blockchain
  - Two nodes find solution to puzzle at roughly the same time, but due to network latency one of them takes longer to reach the peers
  - Double spending attack
- What happens to orphaned blocks?
  - Transactions go back to the queue and will be included in next blocks

Bitcoin Protocol

- Each P2P node runs the following algorithm:
  - New transactions are broadcast to all nodes.
  - Each node (miner) collects new transactions into a block.
  - Each node works on solving proof-of-work (PoW) for its block
    - Use computational resources
  - When a node finds a solution, it broadcasts the block to all nodes.
  - Nodes accept the block only if all transactions are valid (digital signature checking) and coins not already spent (check transactions from public ledger).
  - Nodes express their acceptance by working on creating the next block in the chain
    - If multiple valid blocks are available, choose the longest chain and include transactions from discarded blocks in the queue
    - Include the hash of the accepted block as the previous hash.

Nodes eventually reach global consensus on all transactions

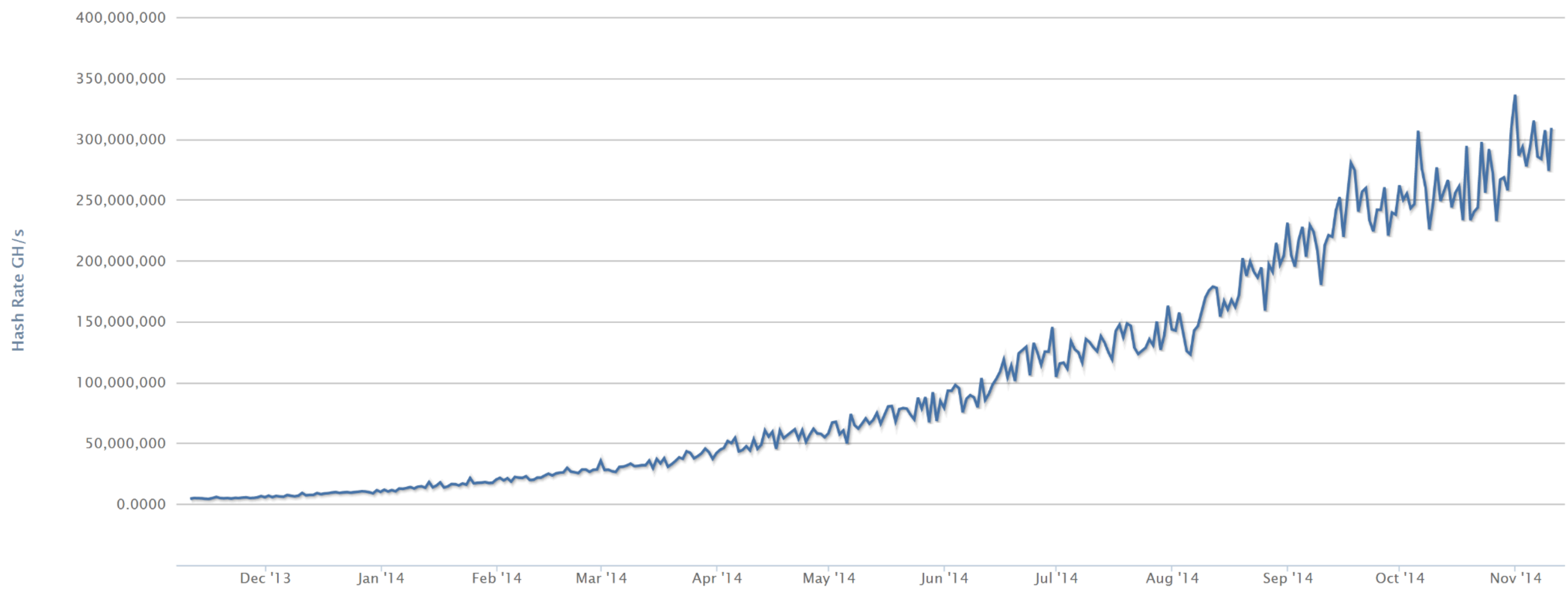
The hardness parameter is periodically changed

- The computing power of the miners **changes**.
- The miners should generate the new block **each 10 minutes** (on average).
- Therefore the hardness parameter **is periodically adjusted** to the mining power
- This happens once each **2016 blocks**.
- For example the block generated on 2014-03-17 18:52:10 looked like this:

> 00000000000000006d8733e03fa9f5e5
> 2ec912fa82c9adfed09fbca9563cb4ce

"Hashrate" = number of hashes computed per second

total hashrate:



Note:

Nov 05 2014 : 283,494,086 GH/s

Nov 05 2013 : 3,657,378 GH/s

≈ 2^{58} hash / second

Eventual consistency

- Consensus doesn't happen right away
- At least 10 mins to verify a transaction
  - Agree to pay
  - Wait for one block (10 mins) for the transaction to go through
  - But, for a large transaction ($$$) wait longer.
    - If you wait longer there will be more blocks mined and higher probability that your transaction is on the consensus chain
    - For large $$$, you wait for six blocks (1 hour) or longer
    - E.g., if a vendor requires 6 confirmations and an attacker controls 10% of the CPU power, the attack will succeed 0.02428% of the time

Main principles

1. It is **computationally hard** to extend the chain (solve puzzle)

2. Once a miner finds an extension he **broadcasts it to everybody**

3. The users will always accept "**the longest chain**" as the valid one

the system incentivizes them to do it

4. **Wait longer** to perform action according to the value of the transaction

Bitcoin security

- Protection again *invalid transactions* (forgery)
  - Cryptographic (digital signature)
- Protection against *modification of blockchain* (remove or modify old transactions)
  - Cryptography (collision-resistant hash functions and digital signatures)
- *Non-repudiation of transactions*
  - Based on blockchain
- Protection against *double spending*
  - Enforced by consensus (correct majority)
  - One of the transactions (either one) will be eventually accepted
- Protection against *Sybil attacks*
  - PoW cryptographic puzzles
  - Assume that adversary does not control majority of CPU resources

What needs to be discussed

1. How is the **trusted bulletin-board** maintained in decentralized manner?
2. How to prevent attackers to obtain majority ?
3. How are users incentivized to be honest?
4. What is the syntax of the transactions?

How are the miners incentivized to participate in this game?

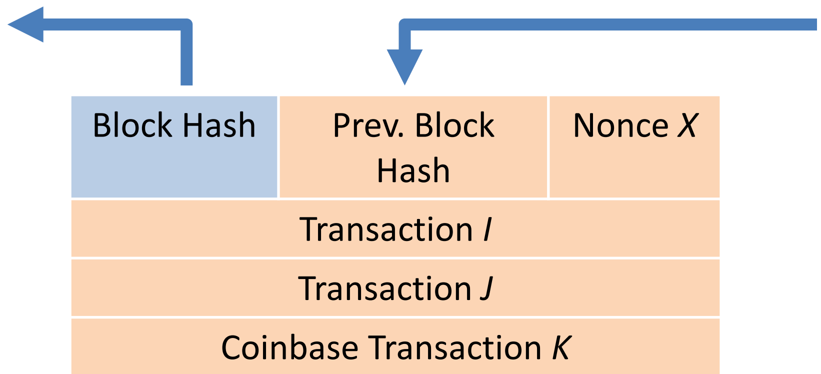
Short answer: they are paid (in Bitcoins) for this

Incentives

- Transactions may include a transaction fee
  - Paid to whoever mines a block that includes the transaction
- New blocks mint new coins
  - Node who wins "mines" a fixed amount of coins as a prize
  - Called a coinbase transaction
  - The only way to generate new coins in the system
- Values of new coins
  - for the first **210,000** blocks: **50 BTC**
  - for the next **210,000** blocks: **25 BTC**
  - for the next **210,000** blocks: **12.5 BTC**,
  - **Note**: $\mathbf{210{,}000 \cdot (50 + 25 + 12.5 + \cdots) \to 21{,}000{,}000}$

Fixed number of blocks in the system

Coinbase Transactions



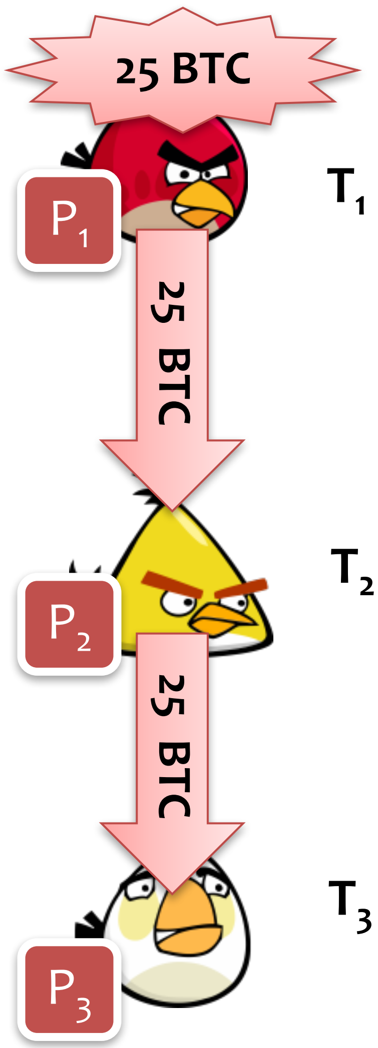
- Generated upon successful mining and included in block chain
- Node will get the reward only if this transaction is on the consensus branch (longest chain)
  - Users are incentivized to mine the longest chain
- Elegantly solves several problems
  - Where do bitcoins come from?
  - How are they minted?
  - Who gets newly minted coins?

What needs to be discussed

1. How is the **trusted bulletin-board** maintained in decentralized manner?
2. How to prevent attackers to obtain majority ?
3. How are users incentivized to be honest?
4. What is the syntax of the transactions?

Transaction syntax – simplified view

We say that **T_3 redeems T_2**



in the "mining process"

T_1 = (User P_1 creates **25 BTC**)

$[T_2]$

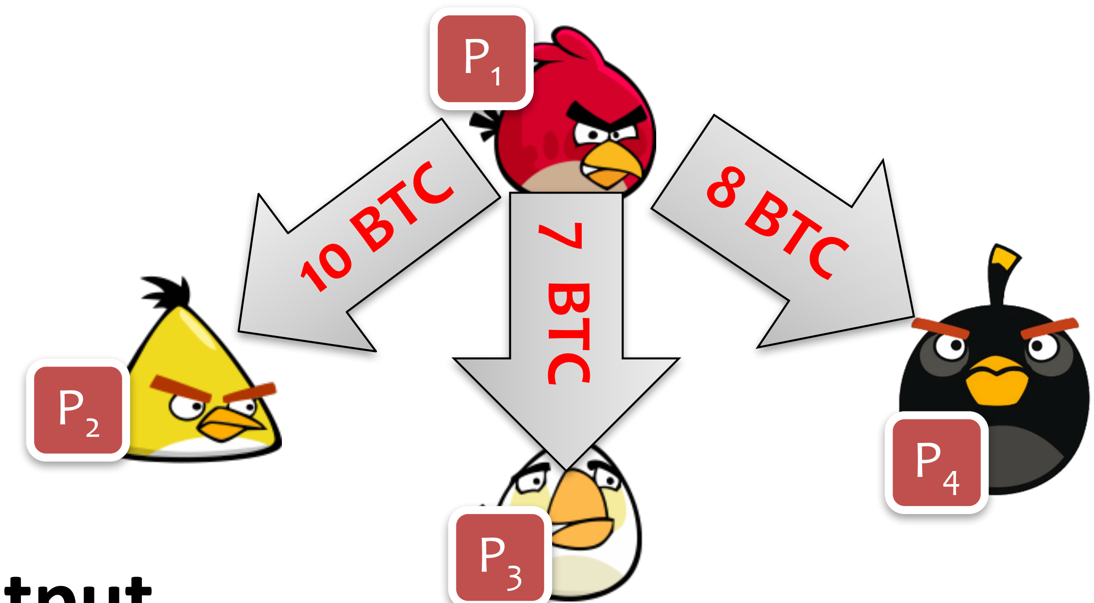
"value of T_2"

T_2 = (User P_1 sends **25 BTC** from T_1 to P_2 | **signature of P_1 on $[T_2]$**)

$[T_3]$

T_3 = (User P_2 sends **25 BTC** from T_2 to P_3 | **signature of P_2 on $[T_3]$**)

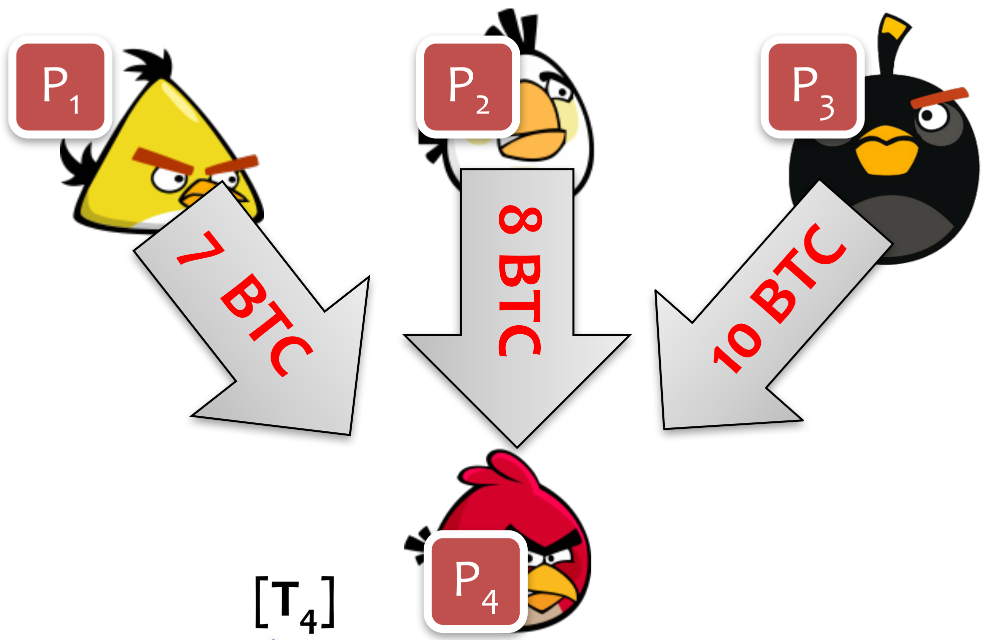
How to “divide money”?



Multi-output transactions: $[T_2]$

T_2 = (User **P_1** sends **10 BTC** from **T_1** to user **P_2 ,**
User **P_1** sends **7 BTC** from **T_1** to user **P_3 ,**
User **P_1** sends **8 BTC** from **T_1** to user **P_4**
signature of P_1 on $[T_2]$)

Multiple inputs



$[T_4]$

T_4 =

(User **P_1** sends **10 BTC** from **T_1** to user **P_4** ,	signature of P_1 on $[T_4]$,
User **P_2** sends **7 BTC** from **T_2** to user **P_4** ,	signature of P_2 on $[T_4]$,
User **P_3** sends **8 BTC** from **T_3** to user **P_4**	signature of P_3 on $[T_4]$)

all signatures need to be valid!

Time-locks

It is also possible to specify time **t** when a transaction becomes valid.

$[T_2]$

T_2 = (User P_1 sends **25 BTC** from T_1 to P_2 if time **t** has passed) | **signature of P_1 on $[T_2]$)**

measured in:

- **real time**, or
- **blocks**.

Generalizations

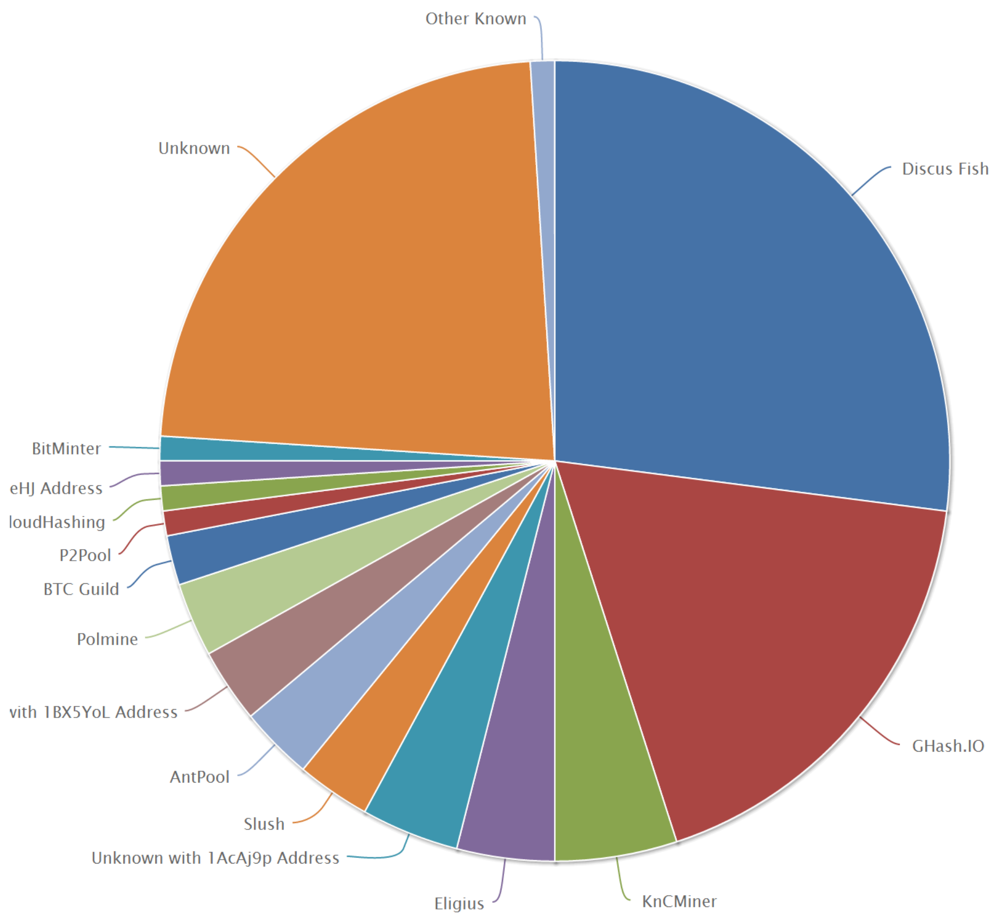
1. All these features can be combined.
2. The total value of **in-coming transactions** can be larger that the value of the **out-going transactions**.

   (the difference is called a "**transaction fee**" **and goes to the miner**)

Popular mining pools

Miners create cartels called **mining pools**

This allows them to reduce the variance of their income



The general picture

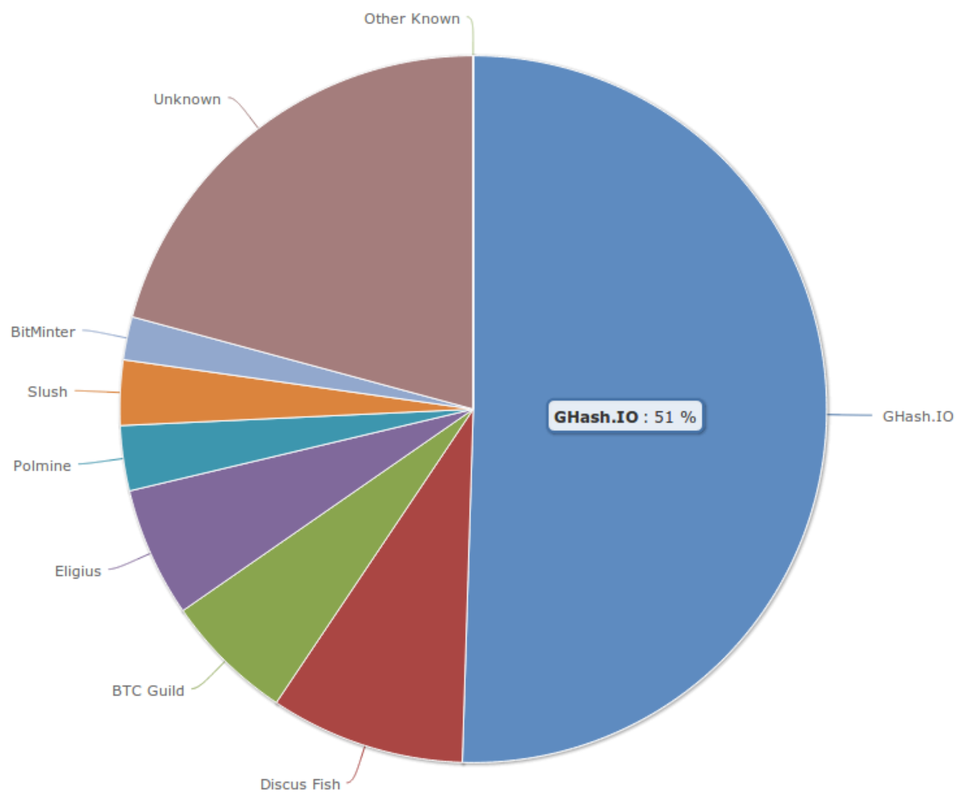
The mining pool is **operated centrally**.

Some of the mining pools **charge fees for their services**.

Tricky part: how to prevent cheating by miners? How to reward the miners?

June 2014

Ghash.io got > 50% of the total hashpower.



Then this percentage went down...

Conclusion

1. People want "cryptocurrencies".
2. Bitcoin has some **important weaknesses**, new ideas are needed.
3. Tricky **security model**.
4. Bitcoin ideas that are interesting on their own:
   a) Consensus based on PoW
   b) Financial incentives
5. Community actively working on other crypto-currencies
   – Different PoW models (memory or storage bound)
   – Improved anonymity levels

Acknowledgement

Some of the slides and slide contents are taken from
http://www.crypto.edu.pl/Dziembowski/teaching
and fall under the following:

We have also used slides from Prof. Dan Boneh online cryptography course at Stanford University:

http://crypto.stanford.edu/~dabo/courses/OnlineCrypto/