Introduction

Stefan Dziembowski
University of Warsaw

Goal of this tutorial

Provide **introduction to the cryptographic currencies and Blockchain** (starting with Bitcoin).

Our main focus: **conceptual aspects** of this field, **new research directions**, and **applications**.

Disclaimer: we omit or simplify many technicalities.

Outline

Today

- **Lecture 1** (9:00 – 10:30) **Introduction**

break

- **Lecture 2** (11:00 – 12:30) **Mining Pools and Security of Bitcoin**

break

- **Lecture 3** (13:30 – 15:00) **Smart Contracts and Off-Chain Protocols**

break

- **Lecture 4** (15:30 – 16:30) **Smart Contracts and Off-Chain Protocols – continued**

Tomorrow

- **Lecture 5** (9:00 – 10:30) **Smart Contracts and Off-Chain Protocols – continued**

break

- **Lecture 6** (11:00 – 12:30) **Alternative Currencies and Blockchains**

break

- **Lecture 7** (13:30 – 15:00) **Techniques for Obtaining Anonymity**

break

- **Lecture 8** (15:30 – 17:00) **Research Directions and Applications**

Plan

1. Introduction
2. Main design ideas of Bitcoin

In a nutshell

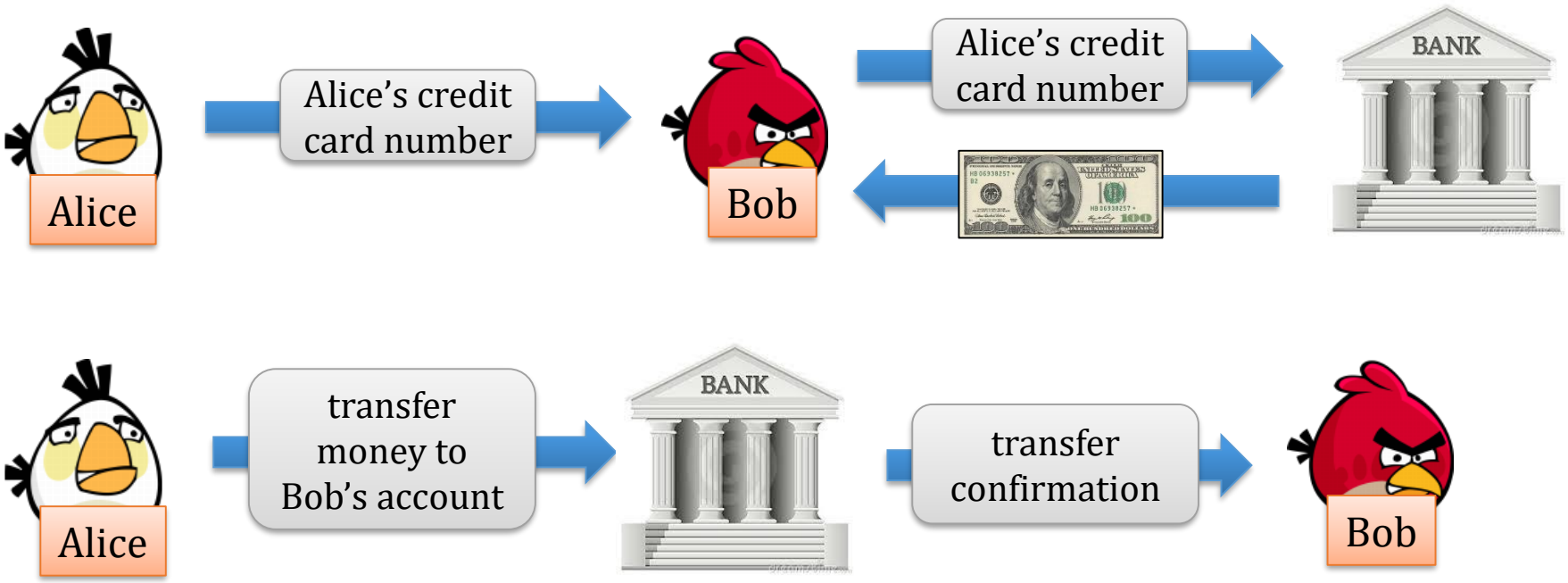
Cryptocurrencies =
"virtual" currencies that can be used for digital payments

Digital vs. paper currencies

Paper:

Digital:

Traditional ways of paying "digitally"

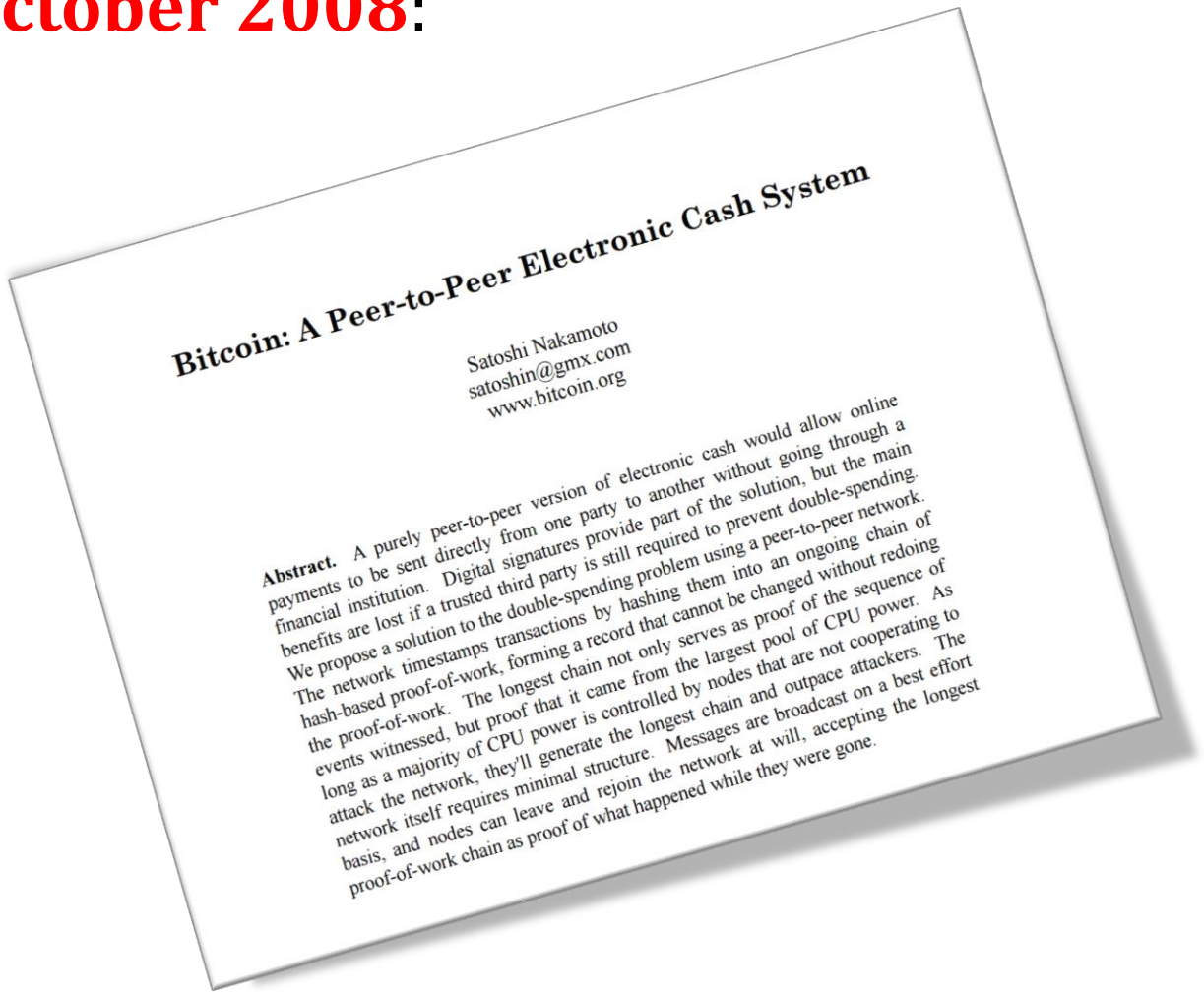


PROBLEMS

1. **trusted server** for each transaction is needed (money doesn't "circulate"),
2. high **transaction fees**,
3. **no anonymity**.

this was the situation until 2008...

then in **October 2008**:

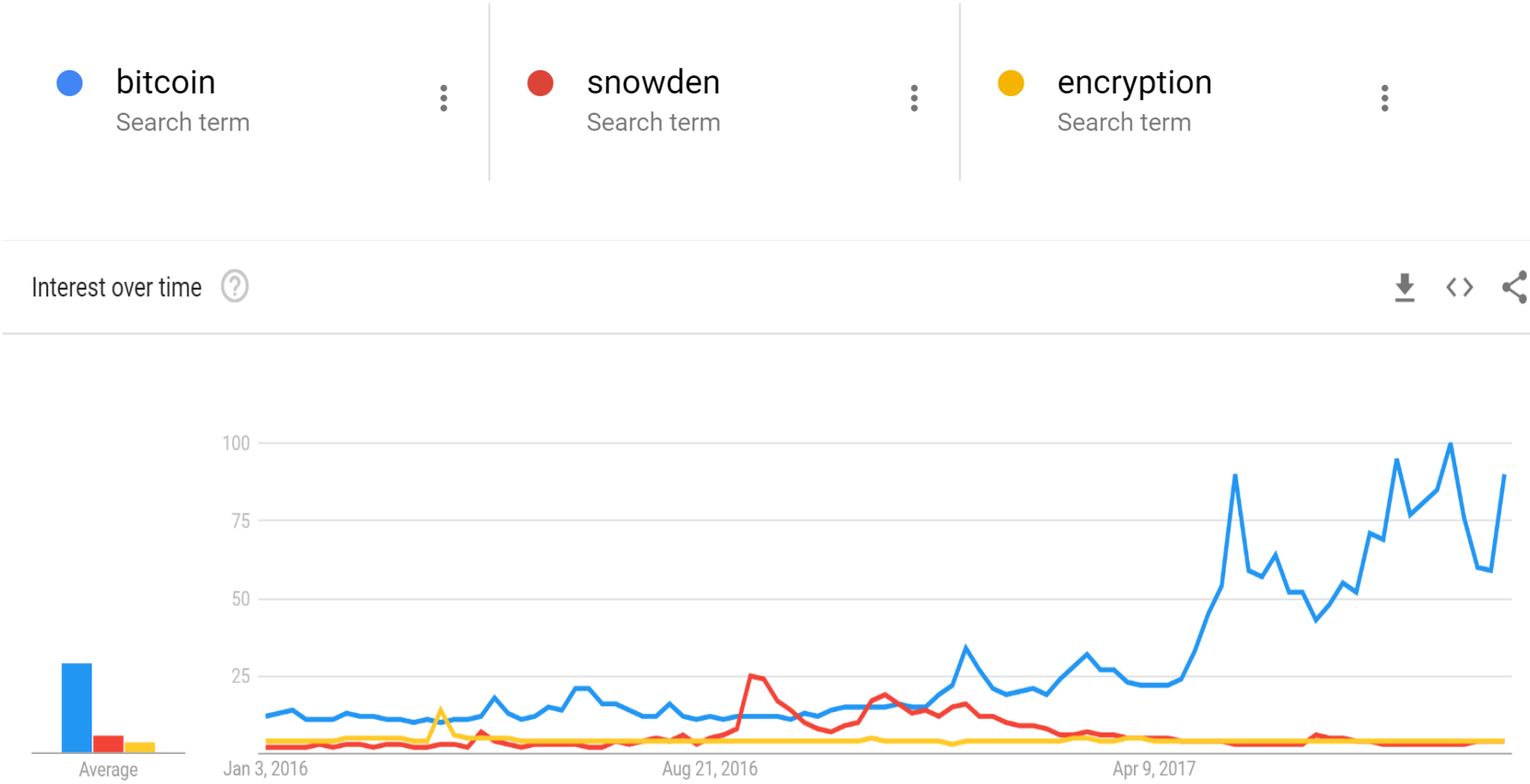


Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Probably one of the most discussed cryptographic technologies ever!



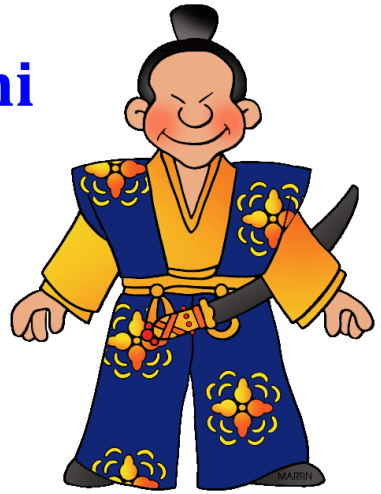
Bitcoin in a nutshell: a "digital analogue" of the paper money

A digital currency introduced by "Satoshi Nakamoto" in 2008.

we will explain it later

Based on the assumption that ``**the majority of the computing power is honest**".

currency unit: **Bitcoin (BTC) 1 BTC = 10^8 Satoshi**



as of Dec 4, 2018:

Market cap ≈ USD 70 billion

1 BTC ≈ USD 4,000 USD

Bitcoin

in Bitcoin:

no trusted server, money circulates

low fees (initially)

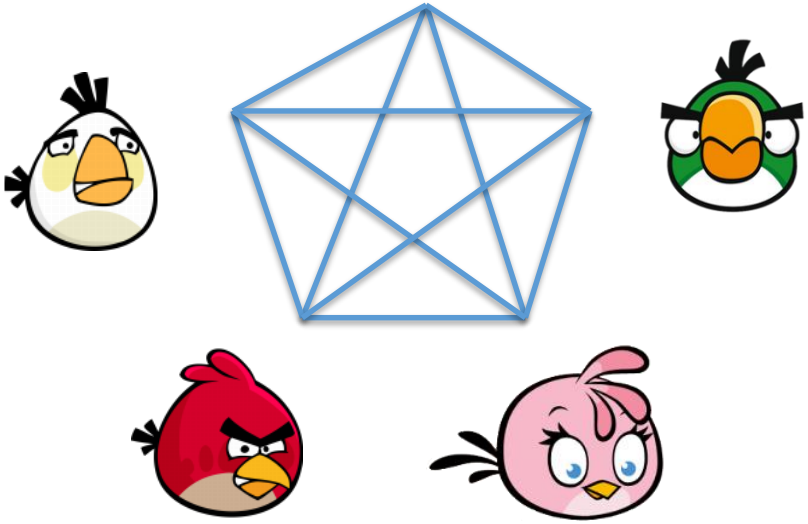
"pseudonymity"

PROBLEMS WITH PREVIOUS APPROACHES

1. **trusted server** is needed (money doesn't "circulate"),
2. high **transaction fees**,
3. **no anonymity**.

“no trusted server” – what does it mean?

“**permissionless**”



everybody can join the system

the users are not “registered” by any authority

they contact each other directly

hence the name: **peer-to-peer network**

“No trusted server”

nobody “**controls the money**”, and therefore:

- The amount of money that will ever be “printed” is fixed (to around 21 mln BTC) → **no inflation**
- The **exchange rate fluctuates**:

Really “no trusted server”?

The client software is written by people who are in power to change the system.

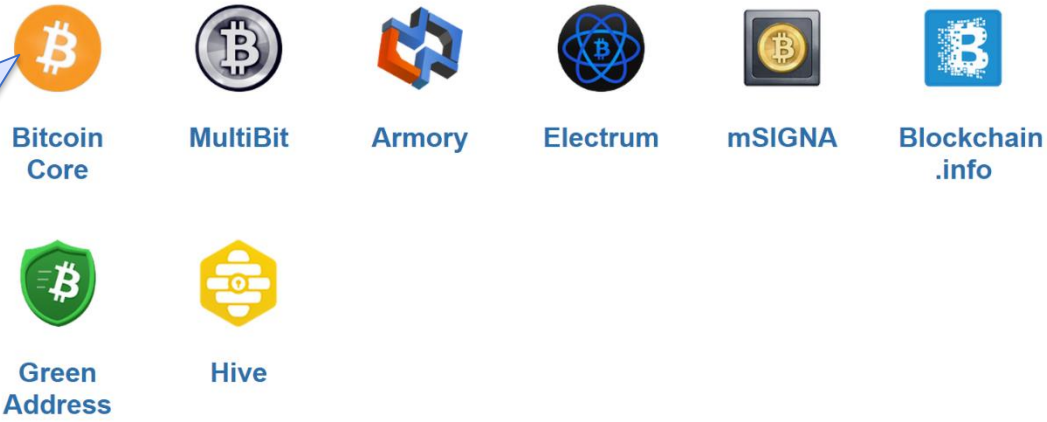
They contain so-called **checkpoints** (more on this later).

For example, this is the list of “desktop clients”:

The most popular client.

(open source)

The developers: Wladimir J. van der Laan, Gavin Andresen, Jeff Garzi, Gregory Maxwell, Pieter Wuille,...



How to update the protocol if there is no governing body?

- Updates have a form of **Bitcoin Improvement Proposals (BIPs)**.

- The Bitcoin community has a **mechanism to vote on BIPs** (weight of the vote **on is proportional to the voter's computing power**),
- the voting **process** is organized centrally (see: github.com/bitcoin/bips):

  ("People wishing to submit BIPs, first should propose their idea or document to the bitcoin-dev@lists.linuxfoundation.orgmailing list. After discussion, please open a PR. After copy-editing and acceptance, it will be published here.")

(we will later talk more about it)

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:

It’s a Ponzi scheme

P. Krugman A. Greenspan

Carlo Pietro Ponzi

The Economist (Nov 1st, 2017)

The Economist

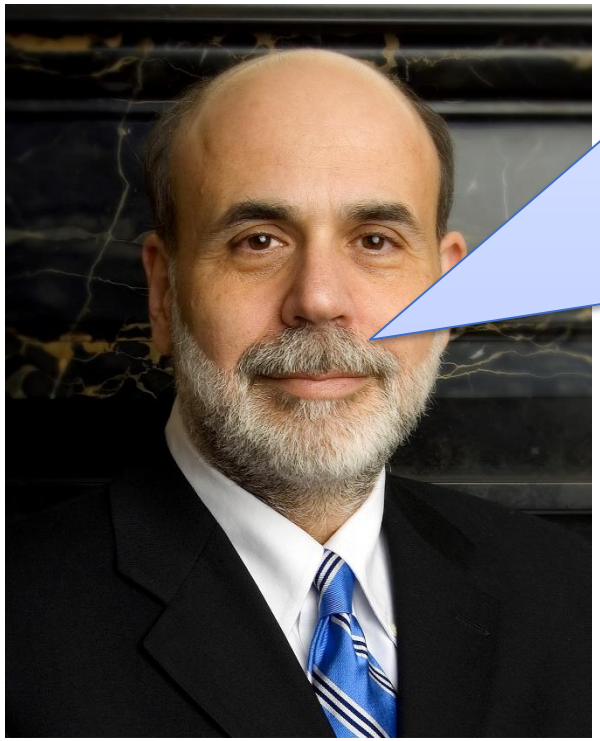
"People are buying Bitcoin because they expect other people to buy it from them at a higher price; the definition of the greater fool theory."

Greater fool theory

The bitcoin bubble

There may be good reasons for buying bitcoin. But the d[illegible] reason at the moment is that it is rising in price

Some economists are more positive



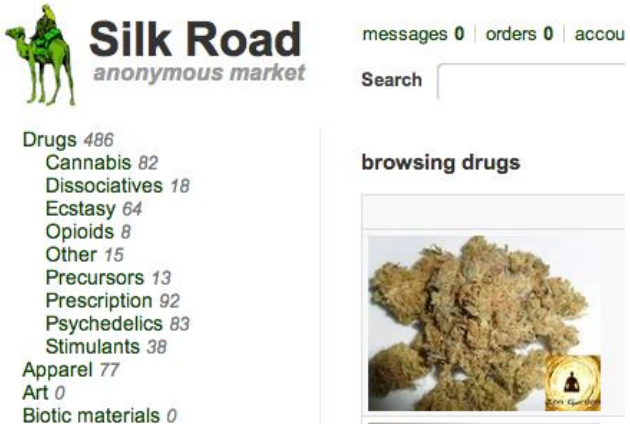
Ben Bernanke

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**.

Why did Bitcoin become so popular (1/2)?

- Ideological reasons (crypto-anarchism).

- Good timing (in 2008 the "quantitative easing" in the US started).



- Seeming anonymity (anonymous enough for trading illegal goods?)

Why did Bitcoin become so popular (2/2)?

- **Low transaction fees.**
- **Hype**?
- Very popular **in some non-democratic countries** (until the government forbids to use it).

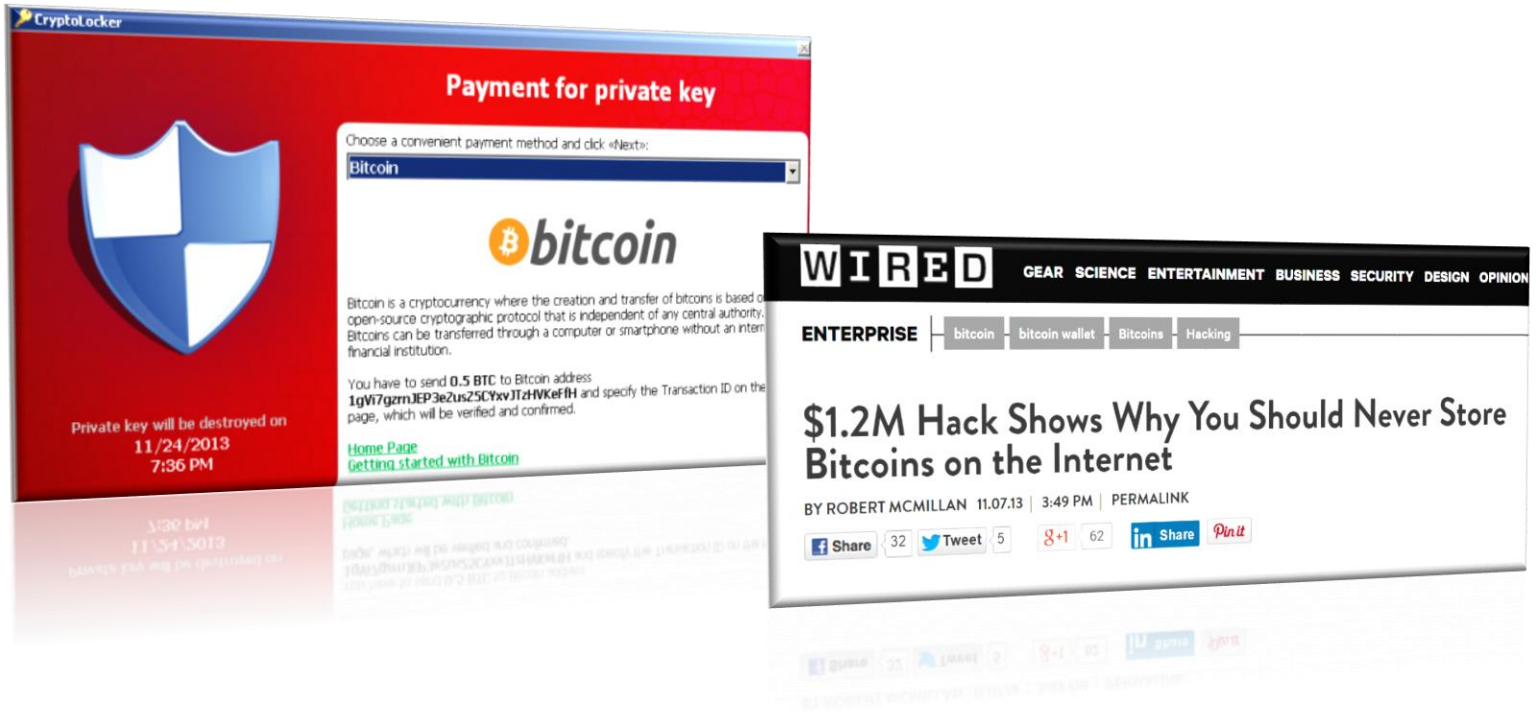
Downsides of decentralization (1/2)

There are no “regulators”…

MtGox (handling **70%** of all Bitcoin transactions) shut down on **Feb 2014** reporting 850,000 bitcoins (≈ **450 million USD**) stolen.

Downsides of decentralization (2/2)

Nobody can reverse transactions, so finally **hackers have good reasons to break into personal computers**.

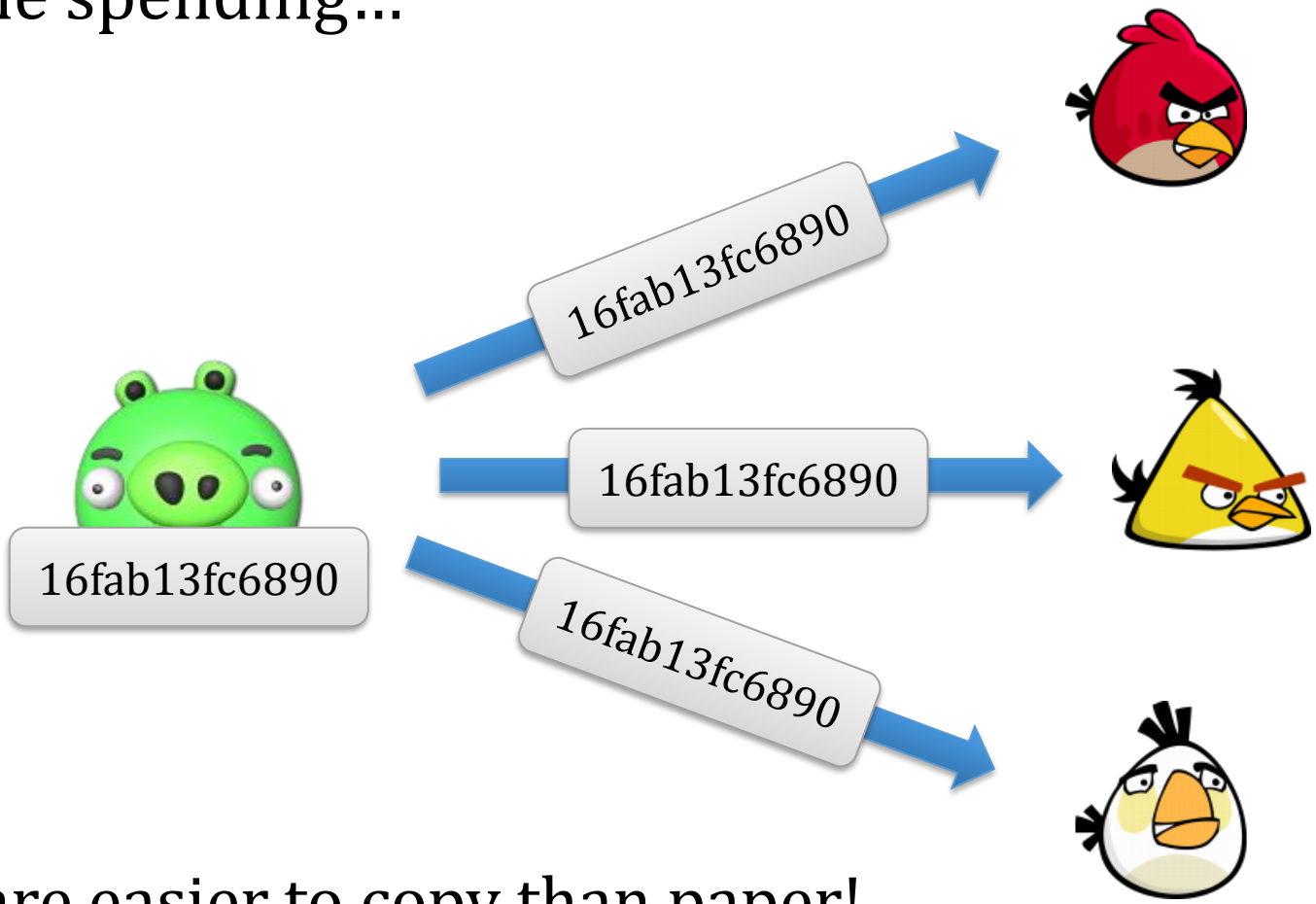


Plan

1. Introduction
2. Main design ideas of Bitcoin

Main problem with the digital money

Double spending…



Bits are easier to copy than paper!

Bitcoin idea (simplified):

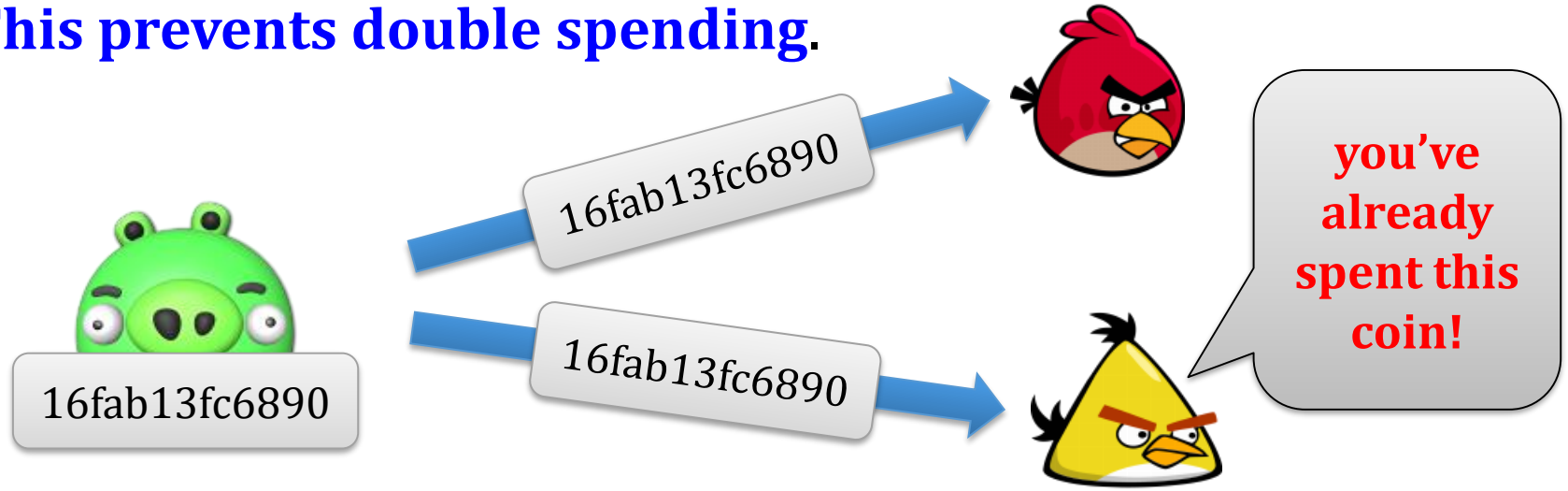
"immutable" "ledger"

The users emulate a **public write-only bulletin-board** containing a list of transactions.

A transaction is of a form:

"User $\mathbf{P_1}$ transfers a coin #16fab13fc6890 to user $\mathbf{P_2}$"

This prevents double spending.

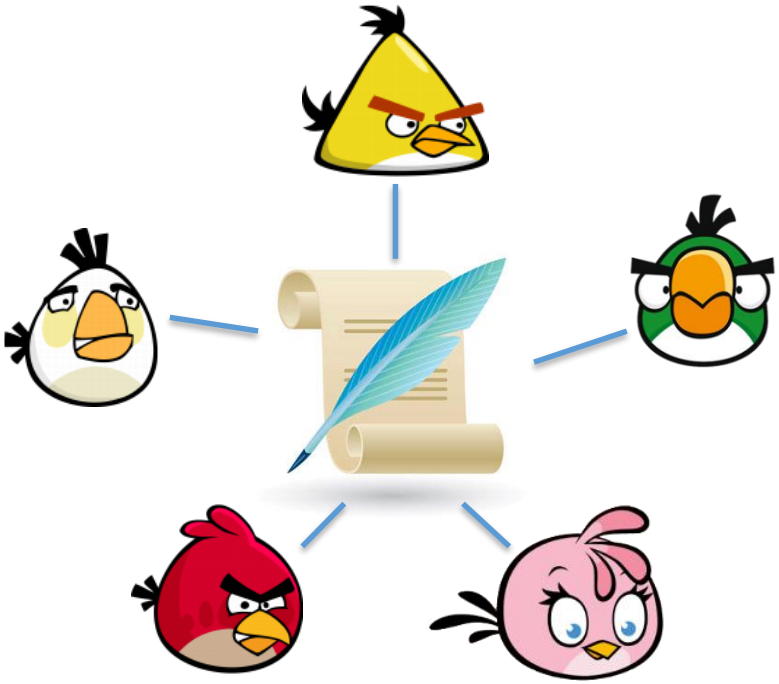


Plan

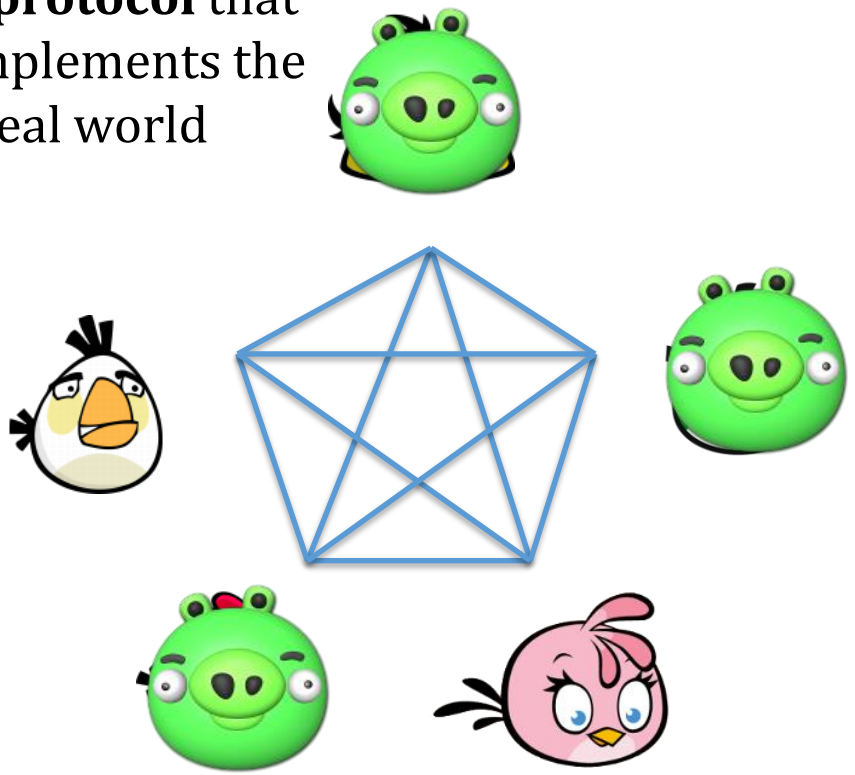
1. Introduction
2. Main design ideas of Bitcoin
    1. How is the ledger maintained?
    2. How are the users identified?
    3. Where does the money come from?
    4. What is the syntax of the transactions?

Ledger emulation

the "ideal" world



a **protocol** that implements the ideal world



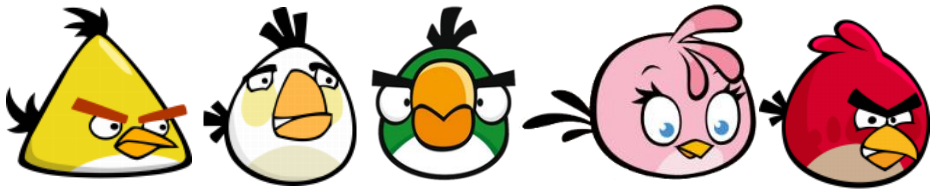
Main difficulty: Some parties can cheat.

An idea

"honest" = they always tell the truth

Assume that the majority of the parties is honest. Then the ledger can be implemented by "voting".

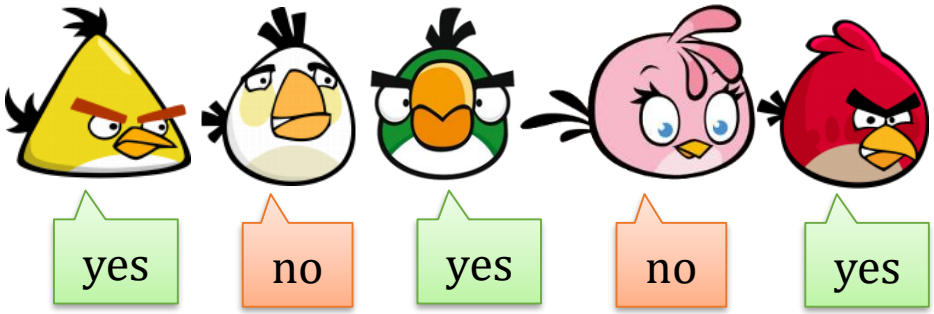
Every transaction is broadcast to all the parties



"Is **this** the correct contents of the ledger?"

dd8bbeabc093b91e4402df4ba...	0.08431821 BTC
54166c365fd6ef4dc22c23e72...	0.6905818 BTC
900852167a13629873ac6defd...	0.11825461 BTC
6e51eb9fbc68bad9b3f62cd4f...	0.00362128 BTC
2842d89b36bc6041c89902cc4...	0.07622 BTC
e3bb90693a84b81384b0719f3...	0.0023 BTC
28a7953700f9dccadf779b194...	0.9998 BTC
008bfc174da83ac895636883c...	2.0698 BTC
a02a15eea695a066a9d2db4f7...	0.30642891 BTC
edb62013b99cb0162e2595fc6...	1.00491631 BTC

⋮



Problem

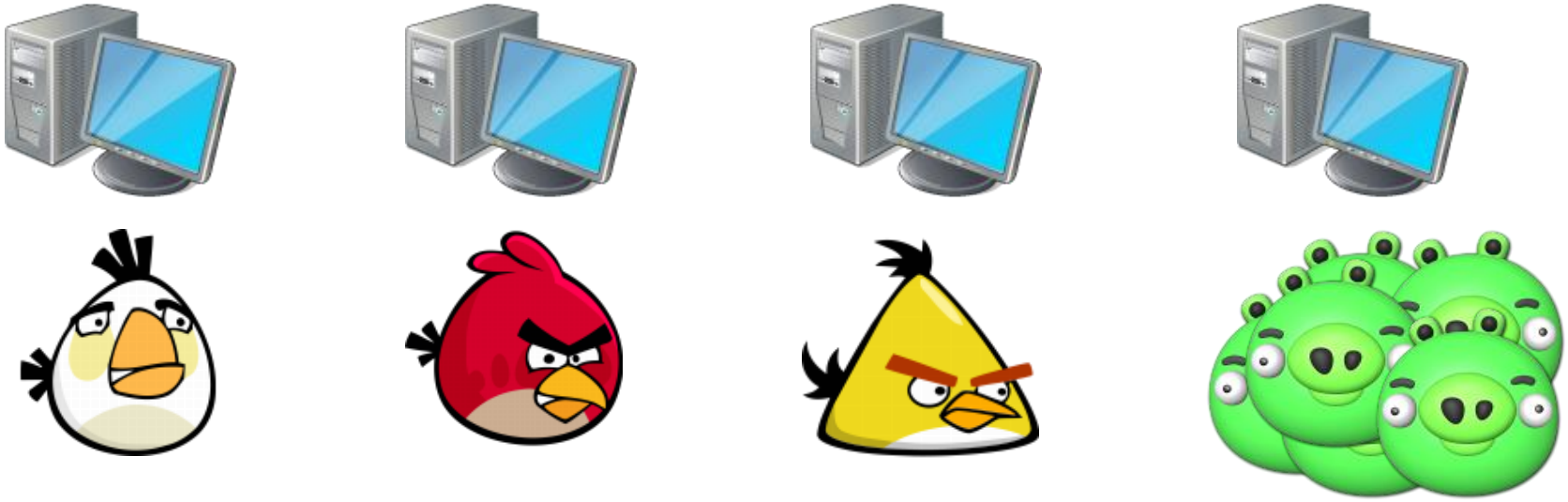
How to define "**majority**" in
a situation where
everybody can join the network?

The Bitcoin solution

Define the “majority” as

the majority of the computing power

Now creating multiple identities does not help!



How is this verified?

Main idea:

a method to "**prove that one did some computational work**" (we explain it in a moment)

- use **Proofs of Work**
- **incentivize** honest users to constantly participate in the process

The honest users can use their **idle CPU cycles**.

Nowadays: often done on **dedicated hardware**.

Main idea

The users participating in the scheme are called the "miners".

They maintain a chain of blocks that is "moderately hard to extend":

the "**genesis block**" created by Satoshi on **03/Jan/2009**

block size < 1MB, which translates to max **7 trans./sec**.

≈ 10 min.

block$_0$ → **block$_1$** → **block$_2$** → **block$_3$** →

transactions from period **1**

transactions from period **2**

transactions from period **3**

Proofs of work

Introduced by **Dwork and Naor** [Crypto 1992] as a countermeasure against spam.

Basic idea:
Force users to do some computational work:

solve a **moderately difficult** "puzzle"

(checking correctness of the solution has to be fast)

We use a PoW build from **hash functions**.

$\mathbf{H}: \{0,1\}^* \rightarrow \{0,1\}^t$
is **a hash function** if it "*behaves like a random function*"

How to construct a PoW?

A building block:

hash functions.

$\mathbf{H}: \{0,1\}^* \rightarrow \{0,1\}^t$
is a hash function if it "*behaves like a random function*"

Practical examples of hash functions: MD5, SHA1, SHA256, SHA3,...

Implemented in every operating system

```
sha256("European Patent Office")=
00627916864 3ddbe1bdecfb41a3a2699828df3fca950728325db33e9d4f2bb7f
sha256("Europeen Patent Office")=
9fb99456acbfeb5ca92ecbf820c7fc86940ac55730d9009813b282fe4075af18
sha256("Europeen Patent Ofice")=
b2f3344cbc36f25e2fbdb565b2cb5a93d33cf86361fee33cb0bdfbd7fcb85dd1
```

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)

s

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

nonce | "hardness parameter

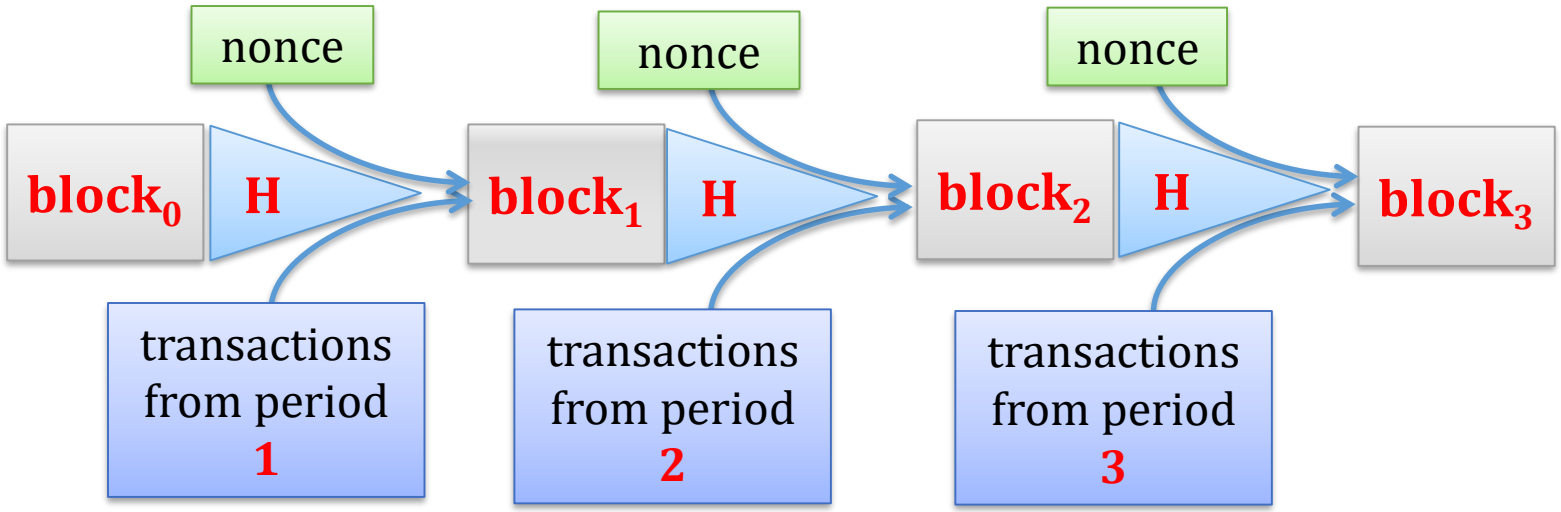
takes time $2^n \cdot$ **TIME(H)**

takes time **TIME(H)**

How are the PoWs used?

H – hash function

more concretely in Bitcoin: **H** is **SHA256**.



Main idea: to extend the chain one needs to find **nonce** such that

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

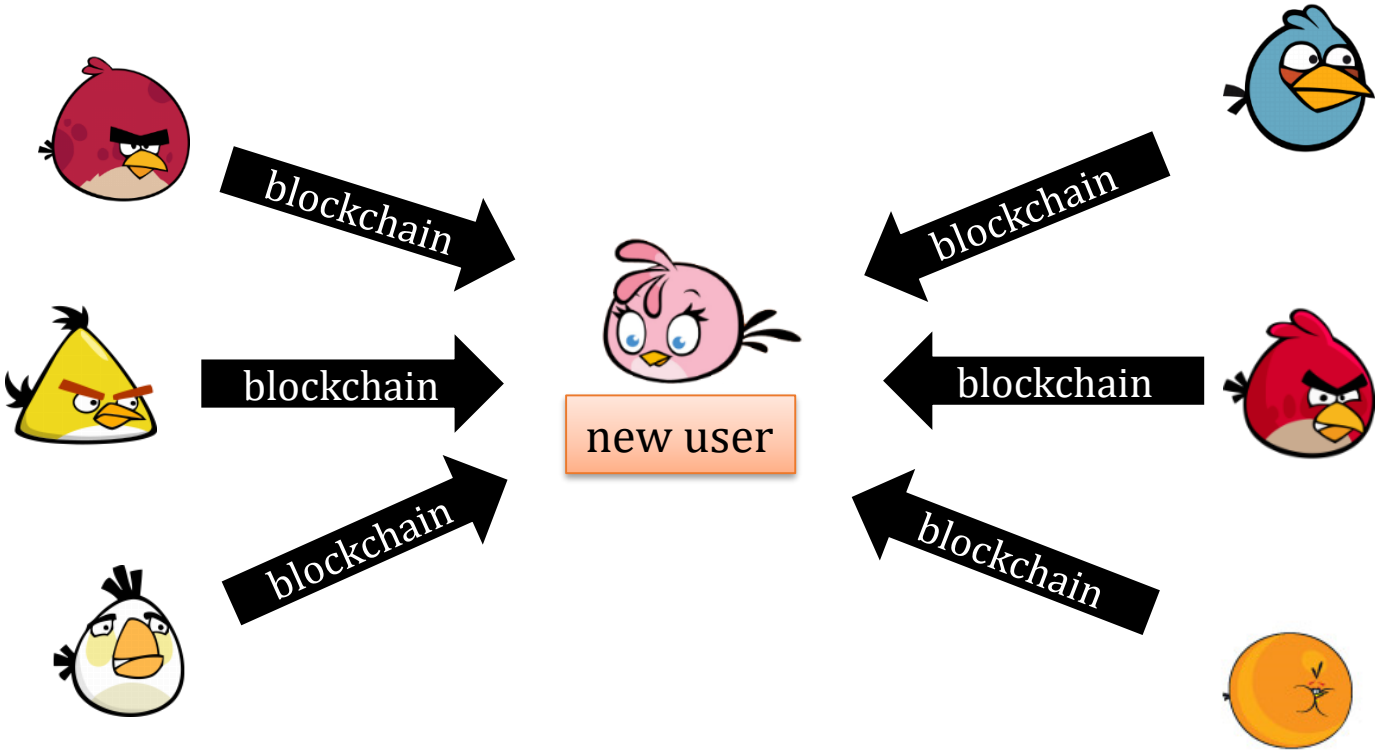
"hardness parameter"

How it looks in real life

Height	Timestamp	Transactions	Size
414902	Jun 5, 2016 5:01:20 PM	386	171361
414901	Jun 5, 2016 4:58:57 PM	304	114339
414900	Jun 5, 2016 4:57:25 PM	1004	428715
414899	Jun 5, 2016 4:50:43 PM	739	384667
414898	Jun 5, 2016 4:45:29 PM	1388	999990
414897	Jun 5, 2016 4:41:19 PM	2187	999945
414896	Jun 5, 2016 4:23:42 PM	2743	998020

Information about the state of the blockchain is propagated in the network

A new user can ask the other users what is the current state of the blockchain.

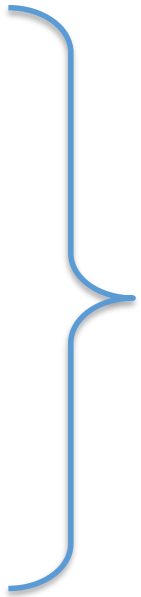


Main principles

1. It is **computationally hard** to extend the chain.

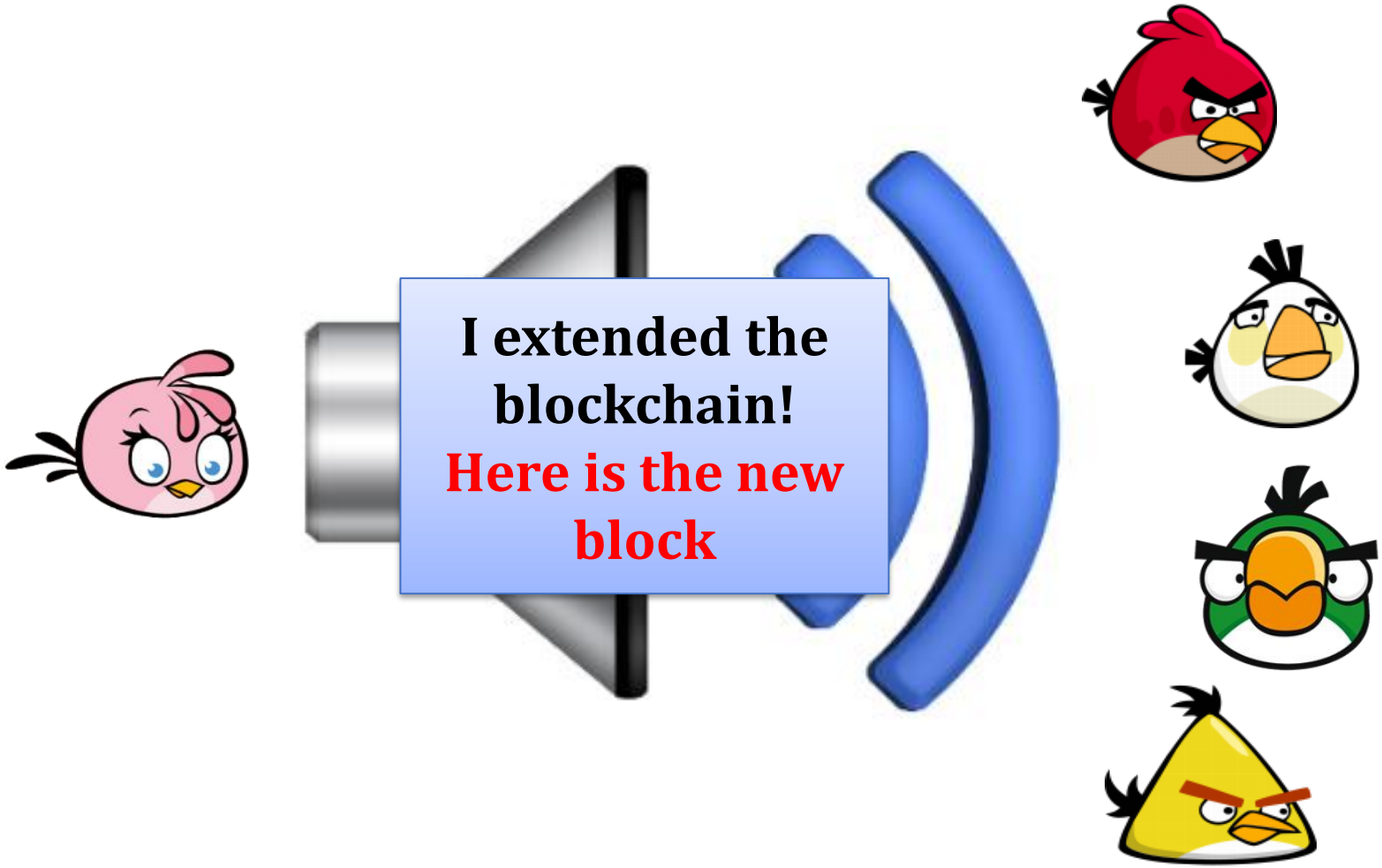
2. Once a miner finds an extension he **broadcasts it to everybody**.

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



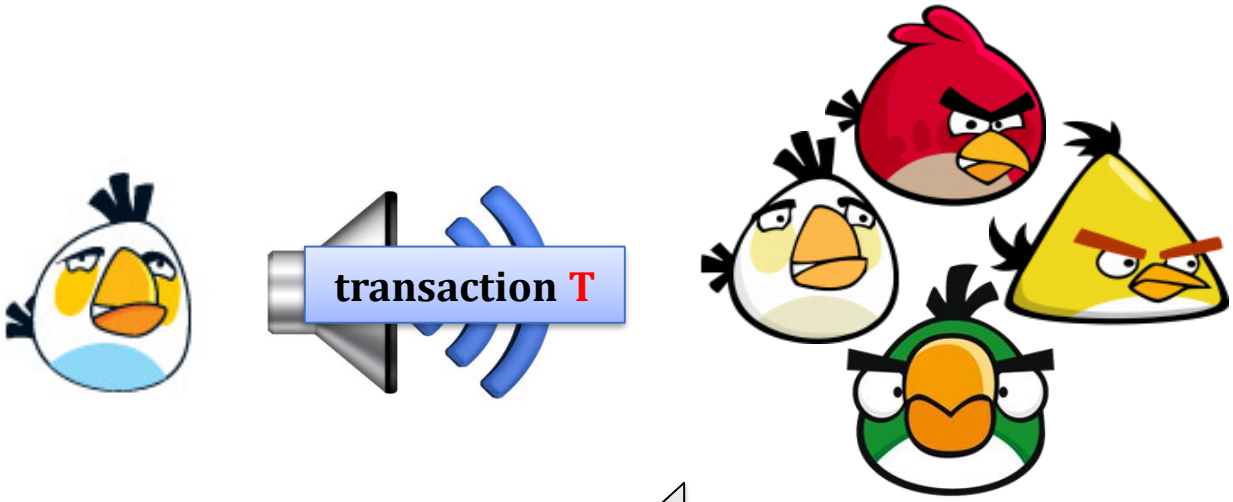
the system incentivizes them to do it

When a new block is mined:



How to post on the board

Just broadcast (over the internet) your transaction to the miners.



And hope they will **add it to the next block.**

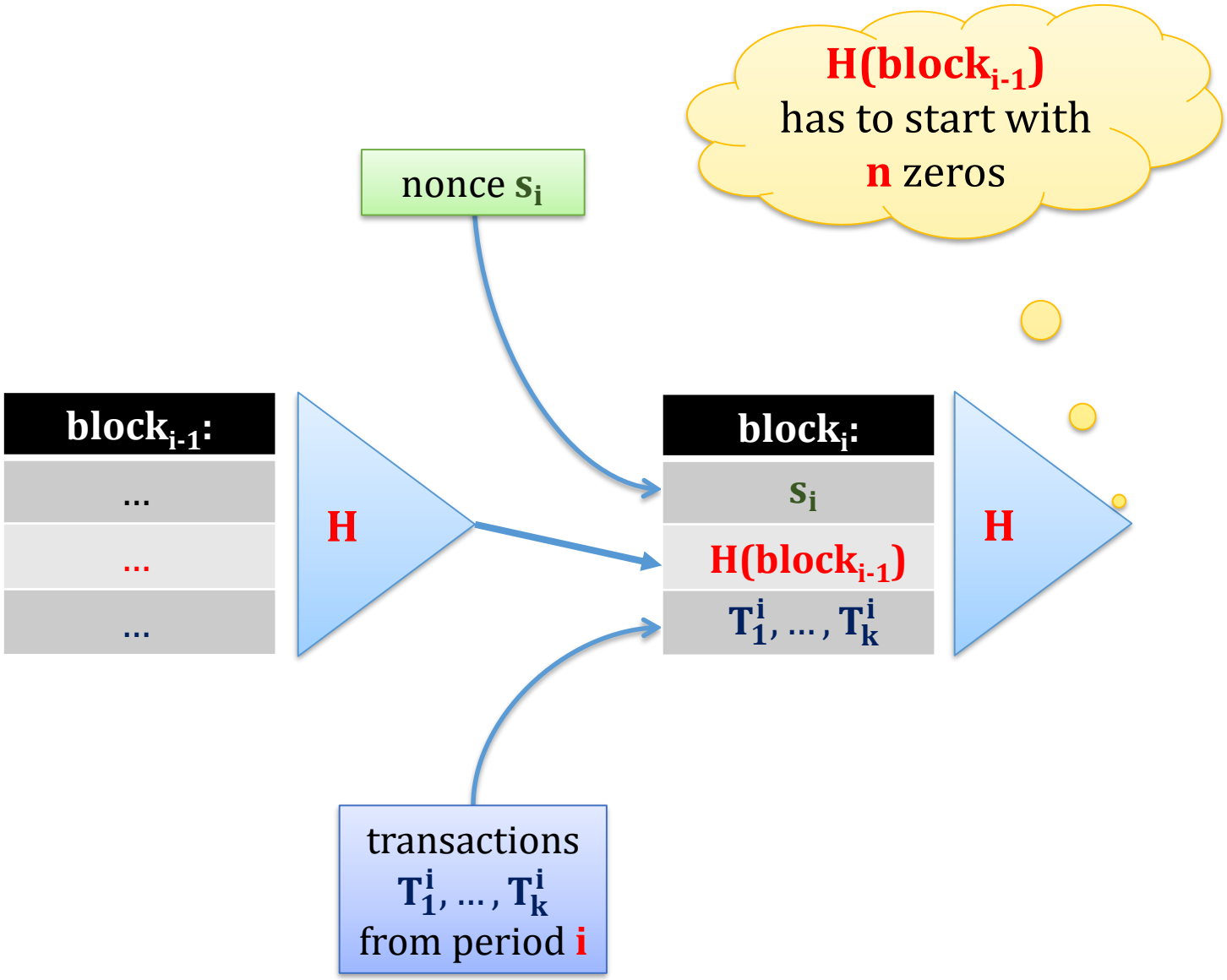
the miners are incentivized to do it.

Important:

They **never add an invalid transaction** (e.g. double spending)

a chain with an invalid transaction is **itself not valid**, so no rational miner would do it.

In more details:



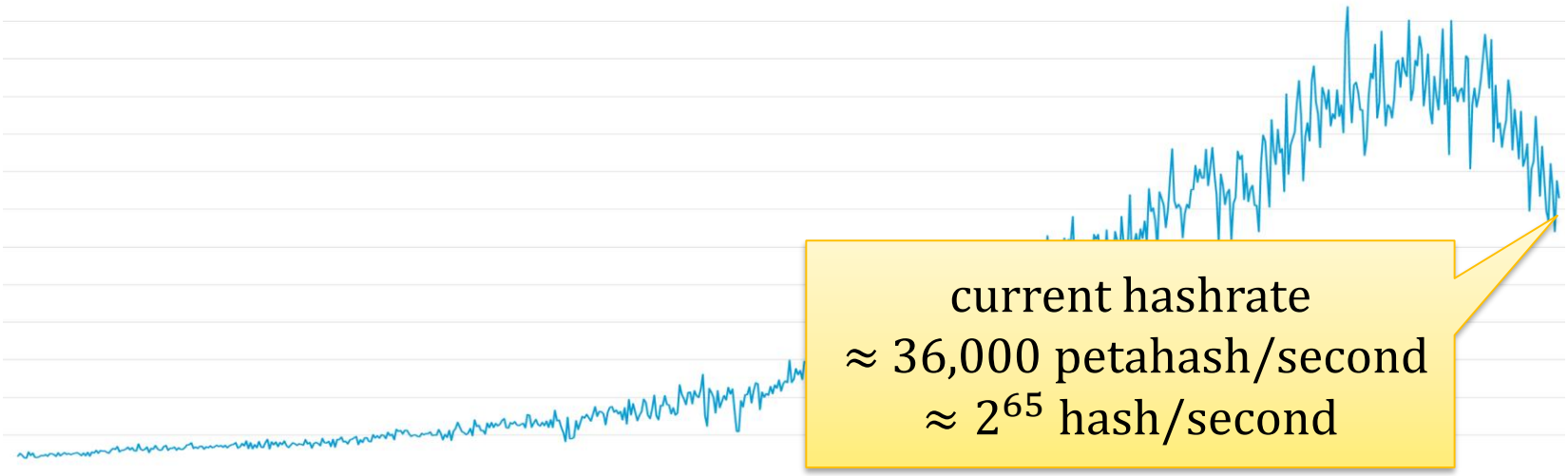
The hardness parameter is periodically changed

- 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**.
- **Important**: the hardness adjustment is **automatic**, and depends on how much time it took to generate last 2016 blocks.

this is possible since every block contains a **time-stamp** produced by the miner who mined it

“Hashrate” = number of hashes computed per second

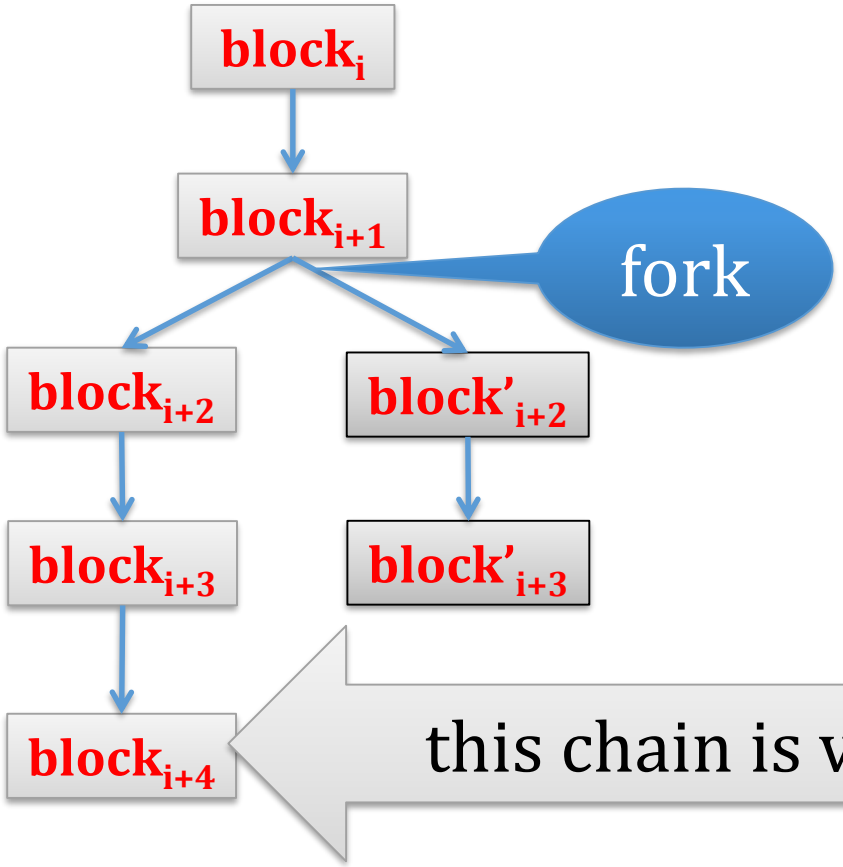
total hashrate over the last **2** years:



Dec 2015:	500 petahash/second
Dec 2016:	2,000 petahash/second
Dec 2017:	12,000 petahash/second
Dec 2018:	36,000 petahash/second

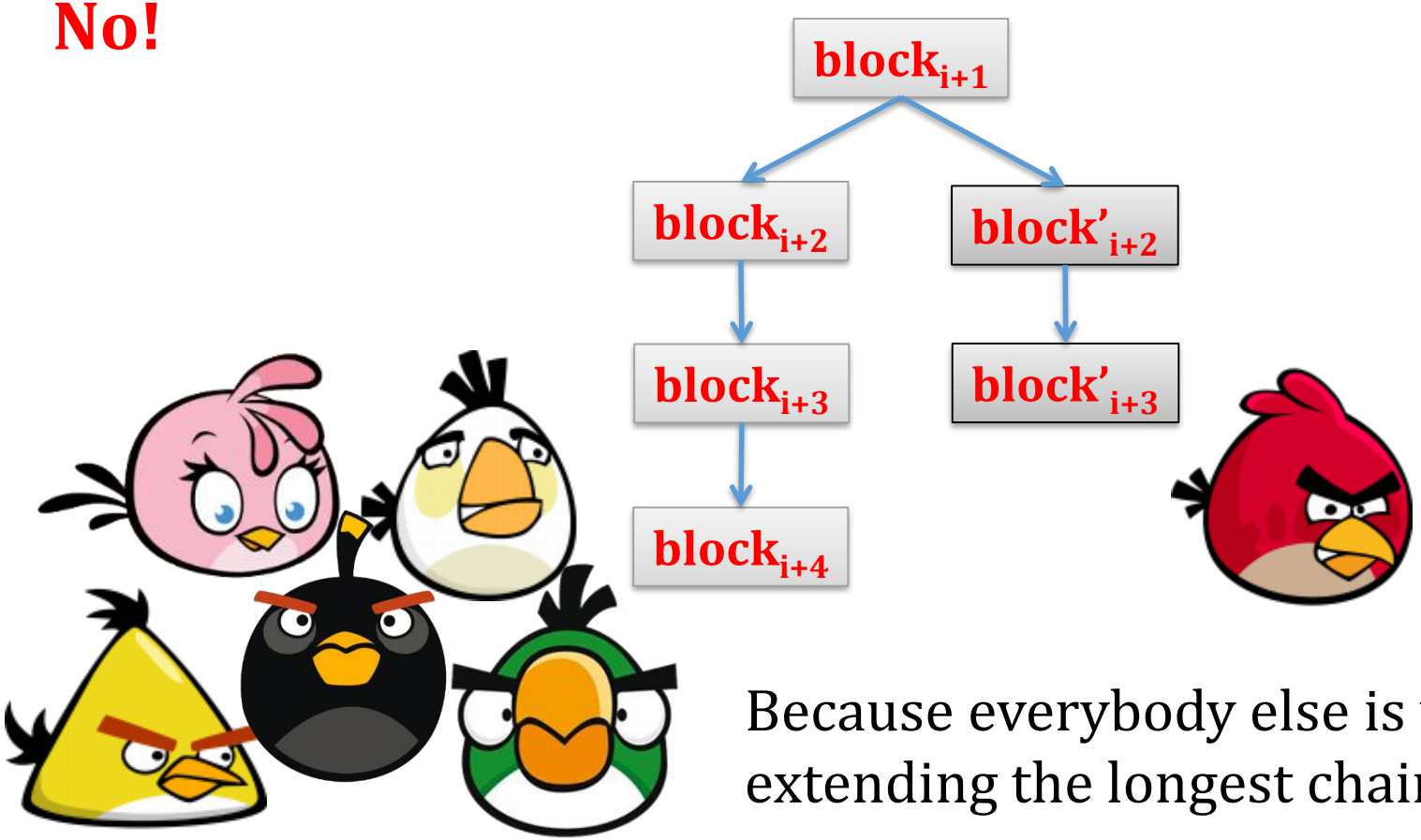
What if there is a “fork”?

For a moment let’s say: the “**longest**” chain counts.



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.

Consequence

The system should quickly **self-stabilize**.

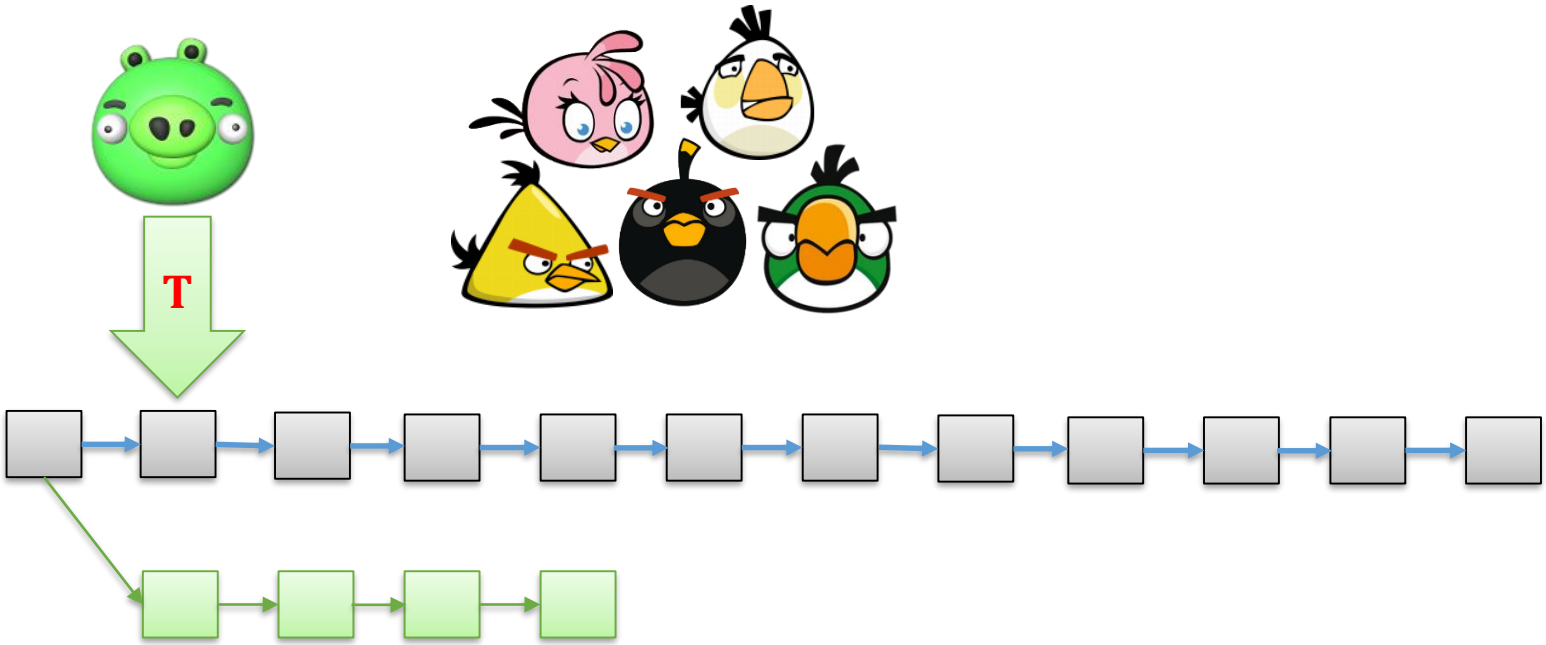
If there is a fork then one branch will quickly die.

Problem: what if your transaction ends up in a "dead branch"?

Recommendation: to be sure that it doesn't happen **wait 6 blocks**.

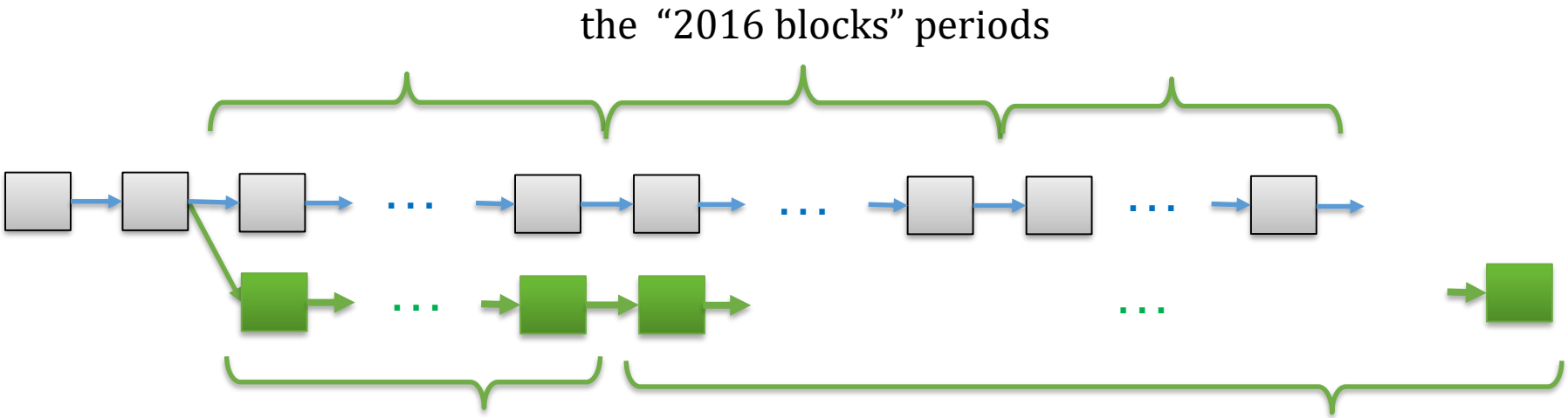
≈ 1 hour

Can transactions be “reversed”?



To reverse a transaction the adversary has to create a “fork in the past”.
This looks very hard if he has a minority of computing power (the honest miners will always be ahead of him).

Since hardness is adjusted thus the following attack might be possible



the adversary forks the chain:

(1)
he computes (secretly) another chain with **fake time-stamps** (indicating that it took a lot of time to produce it)

(2)
the difficulty drops dramatically, so he can quickly produce a chain longer than the valid one, and publish it.

Therefore

In Bitcoin it's not the **longest chain** but the **strongest chain** that matters.

The **strength of each block** is 2^{n}.

n – the hardness parameter in a given period

The **strength of the chain** is the sum of hardnesses of each block in it.

How are the miners incentivized to participate in this game?

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

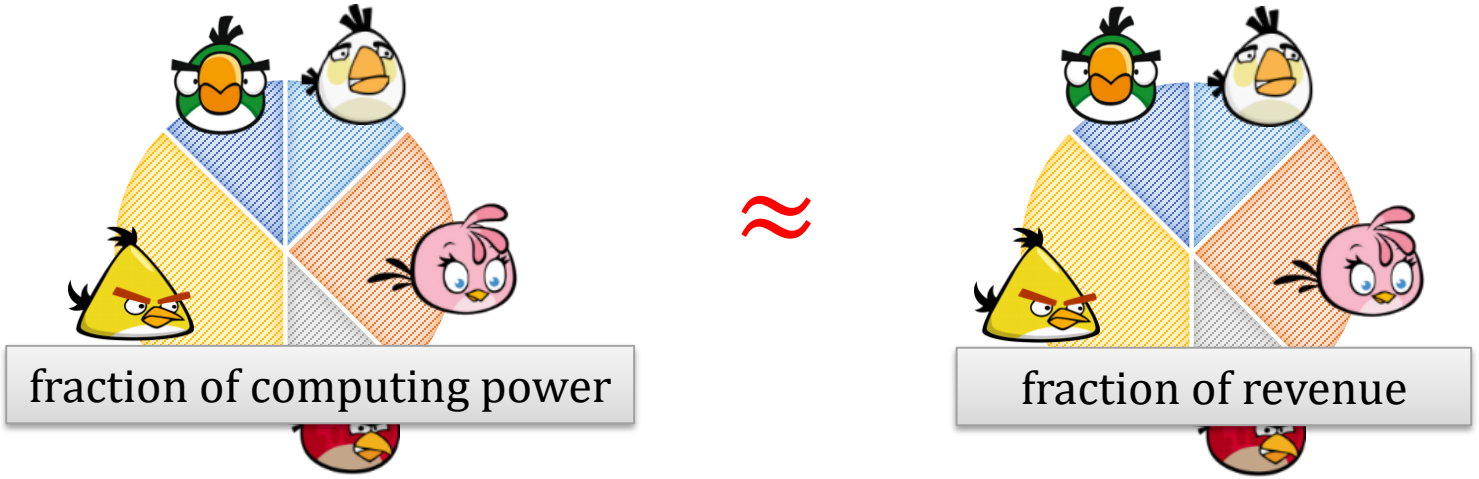
We will discuss it in detail later…

An important feature

Suppose everybody behaves according to the protocol

then:

every miner $\mathbf{P_i}$ whose **computing power is an α_i-fraction** of the total computing power **mines an α_i-fraction of the blocks**.



Intuitively this is because:

$\mathbf{P_i}$'s chances of winning are **<u>proportional to</u>** the number of times $\mathbf{P_i}$ can compute $\mathbf{H}$ in a given time frame.

What is needed to decide which blockchain is valid?

In theory: one needs to know **only**:

- the **initial rules of the game**
- the **genesis block B_0**

This can take several hours.
Note: as of **Dec 2017**:
blockchain's size is ≈ **145 GB**.

Then from many "candidate chains" choose the one that

- **verifies correctly** (starts B_0 and is satisfies all the rules)
- is **the strongest**.

One doesn't even need to have access to the communication history.

In practice: it's not that simple...

we will talk about it in a moment

Freshness of the genesis block

I didn't know the genesis block before Bitcoin was launched (**Jan 3, 2009**)

Here is a heuristic "proof":

Block$_0$ contained a hash of a title from a front page of the London Times on **Jan 3, 2009**

A recent paper that shows how to generate the genesis block in a distributed way: **[Andrychowicz, D., CRYPTO'15]**.

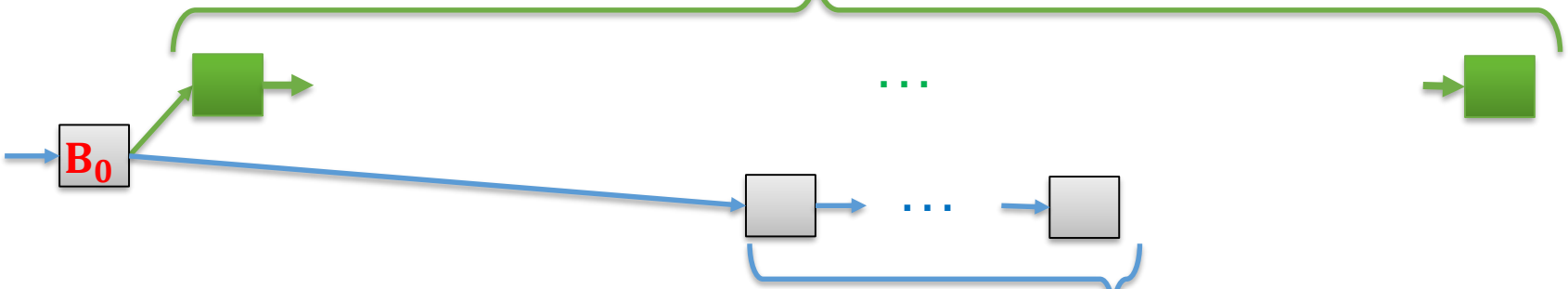
Why is this important?

Otherwise Satoshi could "pre-mine"

(mostly a theoretical threat today):

(1)
secretly start mining in **1980**, produce a very strong chain

(3)
On **Jan 03, 2010** publish your secret chain



(2)
honest miners start working on **Jan 03, 2009**.
They have **more computing power** but **less time**.
So, **after 1 year** their chain is still **weaker** than the one of Satoshi.

Checkpoints

Checkpoint – old block hash **hardcoded into Bitcoin software**.

From the **theoretical** point of view: ***not* needed**.

Moreover: they go against the "decentralized" spirit of Bitcoin.

Still they have some **practical advantages**:

- they prevent some **DoS attacks** (flooding nodes with unusable chains)
- they prevent attacks involving **isolating nodes** and giving them fake chains,
- they can be viewed as an **optimization** for the initial blockchain download.

Protocol updates

The Bitcoin protocol **can be updated**.

Proposals for the Bitcoin updates can be submitted to the **Bitcoin foundation** in the form of the **Bitcoin Improvement Proposals** (**BIPs**).

Then the foundation puts them at vote.

Only the miners can vote. The votes are included in the mined blocks.

Currently it is required that a proposal gets **a certain percentage *P* of approval in the mined blocks** (over some period of time).

Note: ***P* % of blocks ≈ *P* % of computing power** ("economic majority").

Plan

1. Introduction
2. Main design ideas of Bitcoin
    1. How is the ledger maintained?
    2. How are the users identified?
    3. Where does the money come from?
    4. What is the syntax of the transactions?

How to identify the users in the peer-to-peer networks?

Use the **digital signature schemes**.

≈

digital analogue of the handwritten signatures.

handwritten signatures:

Anna's ID

signs the document

Anna Kowalska

everybody can **verify the signature** by looking at the **Anna's ID** and the **message**

---

digital signatures:

11:b5:73:7c:f9:d9:3f:17:c0:c
b:1a:84:65:5d:39:95:a0:28:2
4:09:7e:ff:a5:ed:d8:ee:26:38:

Anna's private key

matching

04:a0:15:32:a3:c0:90:00:
53:de:60:fb:ef:ef:cc:a5:87:
93:30:15:98:d3:08:b4:1e:

Anna's public key

signs the document

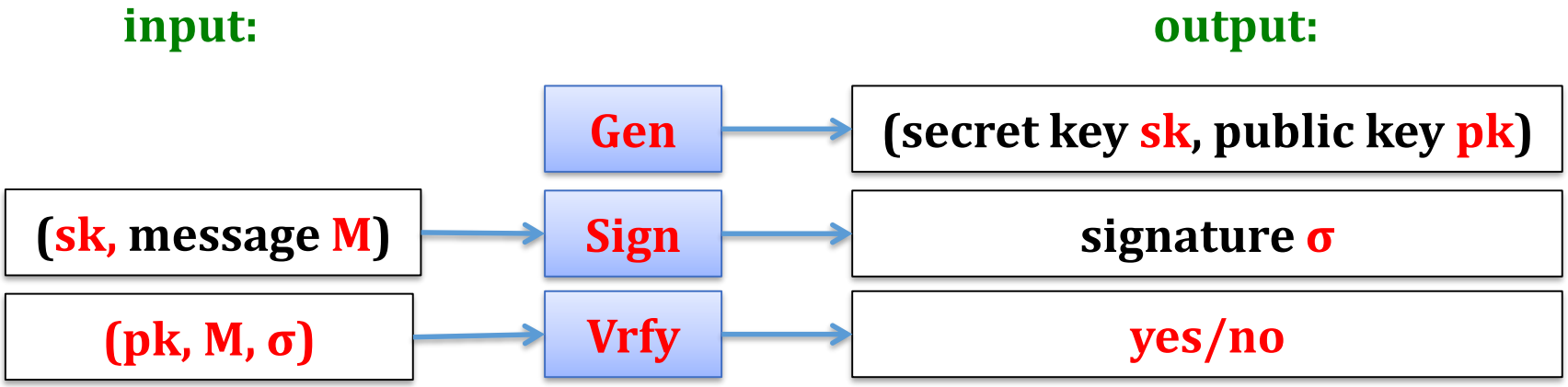
"digital signature"

de:32:a3:c0:
ef:cc:a5:a6

everybody can **verify the signature** by looking at the **Anna's public key** and the **message**

Signature schemes

A **digital signature scheme** consists of algorithms **Gen**, **Sign** and **Vrfy**, where:



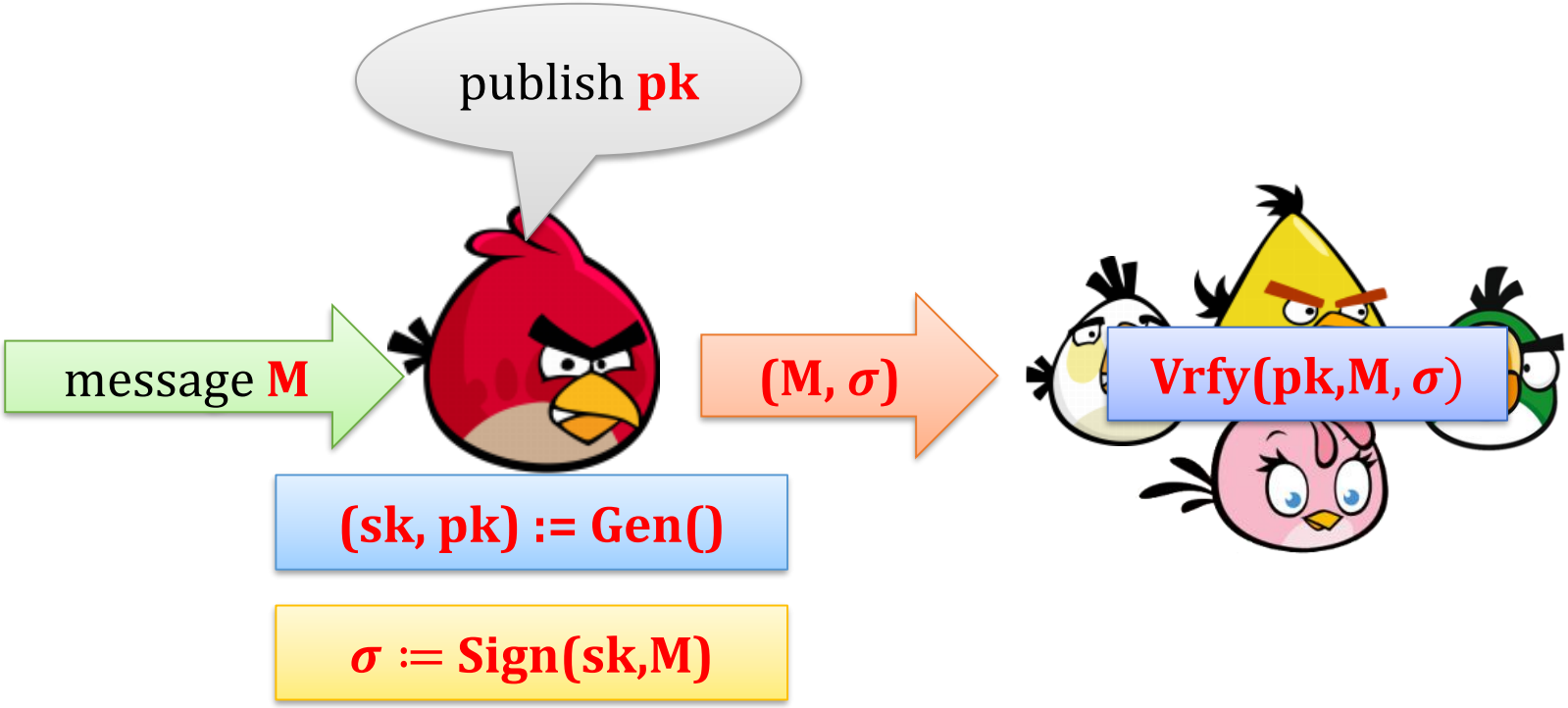
Correctness:

for every **(sk,pk) := Gen()** and every **M** we have
Vrfy(pk,M,Sign(sk,M)) = yes

Security:

"without knowing **sk** it is infeasible to compute **σ** such that
Vrfy(pk,M,σ) = yes"

How to use the digital signatures?



Popular signature schemes

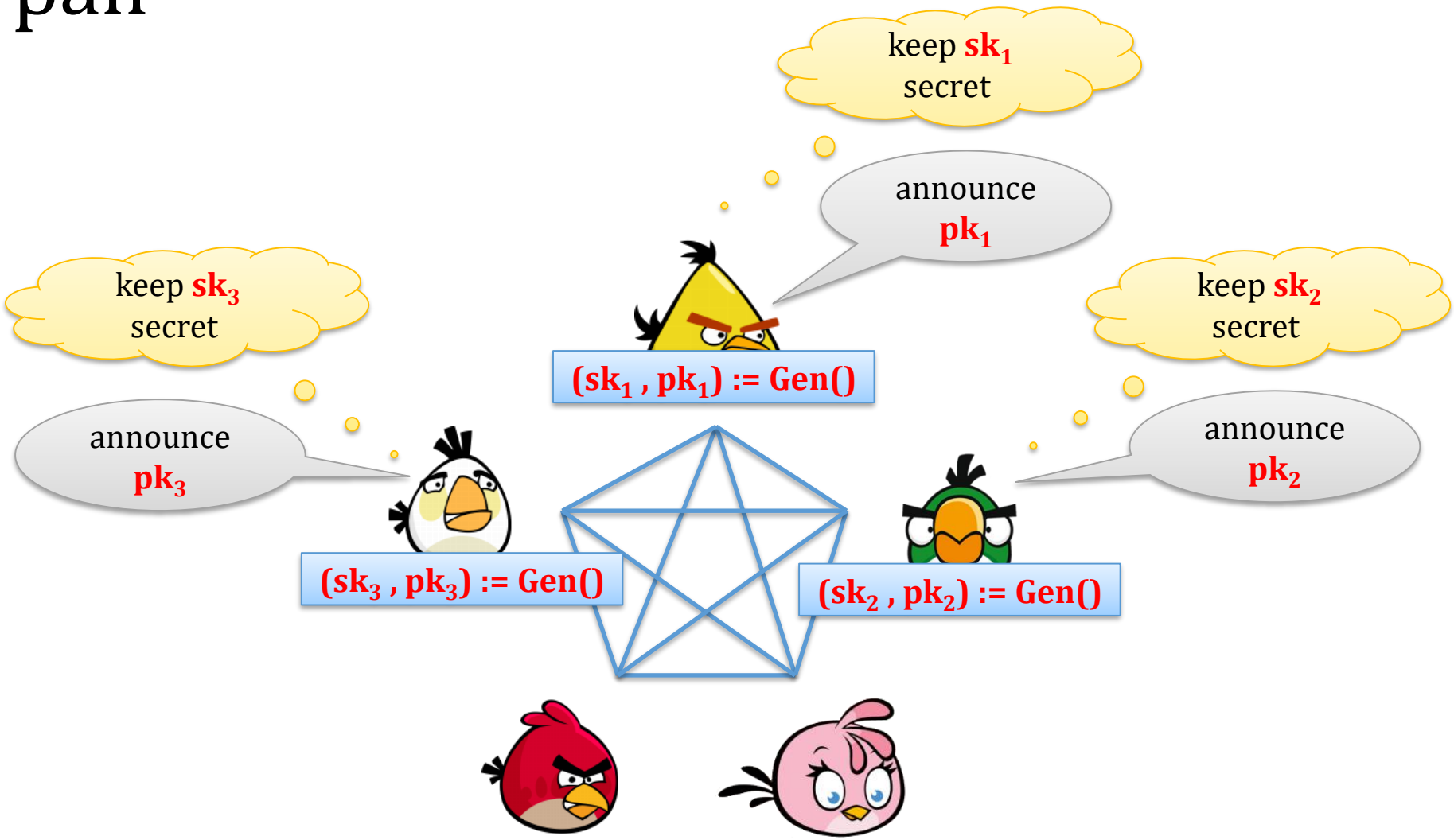
- **RSA** (1970s)
- **ElGamal**
- **DSA**
- **ECDSA**

How to identify the users in the peer-to-peer networks?

We use the digital signature schemes.

The users are identified by their public keys.

Everybody can generate his own key pair



Convenions

Public key in Bitcoin are also called **addresses**.

It is recommended **not** to reuse the addresses.

In other words: for every new transaction one should use a **new address** (mostly: for security and anonymity).

On these slides we often ignore this convention for the sake of simplicity.

Plan

1. Introduction
2. Main design ideas of Bitcoin
    1. How is the ledger maintained?
    2. How are the users identified?
    3. Where does the money come from?
    4. What is the syntax of the transactions?

Where does the money come from?

A miner who finds a new block gets a “reward” in **BTC**:

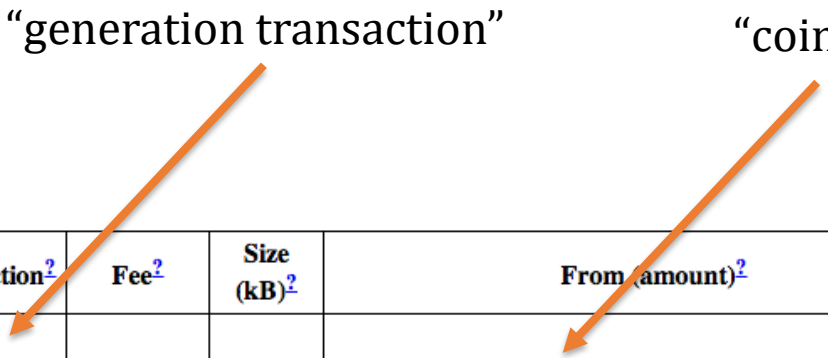
≈ 4 years

- 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**, **current reward**

and so on…

Note: **210,000 · (50 + 25 + 12.5 + ⋯) → 21,000,000**

This is how it looks in detail



Transaction[?]	Fee[?]	Size (kB)[?]	From (amount)[?]	To (amount)[?]
0ac34c9949...	0	0.173	Generation: 25 + 0.05974785 total fees	1KFHE7w8BhaENAswwryaoccDb6qcT6DbYY: 25.05974785
2055f19a51...	0.0002	0.259	1Kpv8JEcWLhUqi4q8dnrwxiaZPKL4KUoeR: 179.9998	1HCukLGfkCfKCryXT73hj2SyVAC9kzRGkC: 105 15zBXYeXbtJ5xs48arouP7BHQu4AQ5xfZa: 74.9996
66815aff01...	0.001	0.258	1dice6DPtUMBpWgv8i4pG8HMjXv9qDJWN: 0.35	15GPjviasjMD8QJvMTs5qYsB8wtQLQGBtP: 0.00175 1HZHBnH2FbHNWieMxAh4xBPfgfuxW15UPt: 0.34725

More details

Each block contains a transaction that **transfers the reward** to the miner.

Advantages:

1. It provides **incentives** to be a miner.
2. It also makes the miners interested in **broadcasting new block** asap.

this view was challenged in:
Ittay Eyal, Emin Gun Sirer
Majority is not Enough: Bitcoin Mining is Vulnerable
(we will discuss it later)

Plan

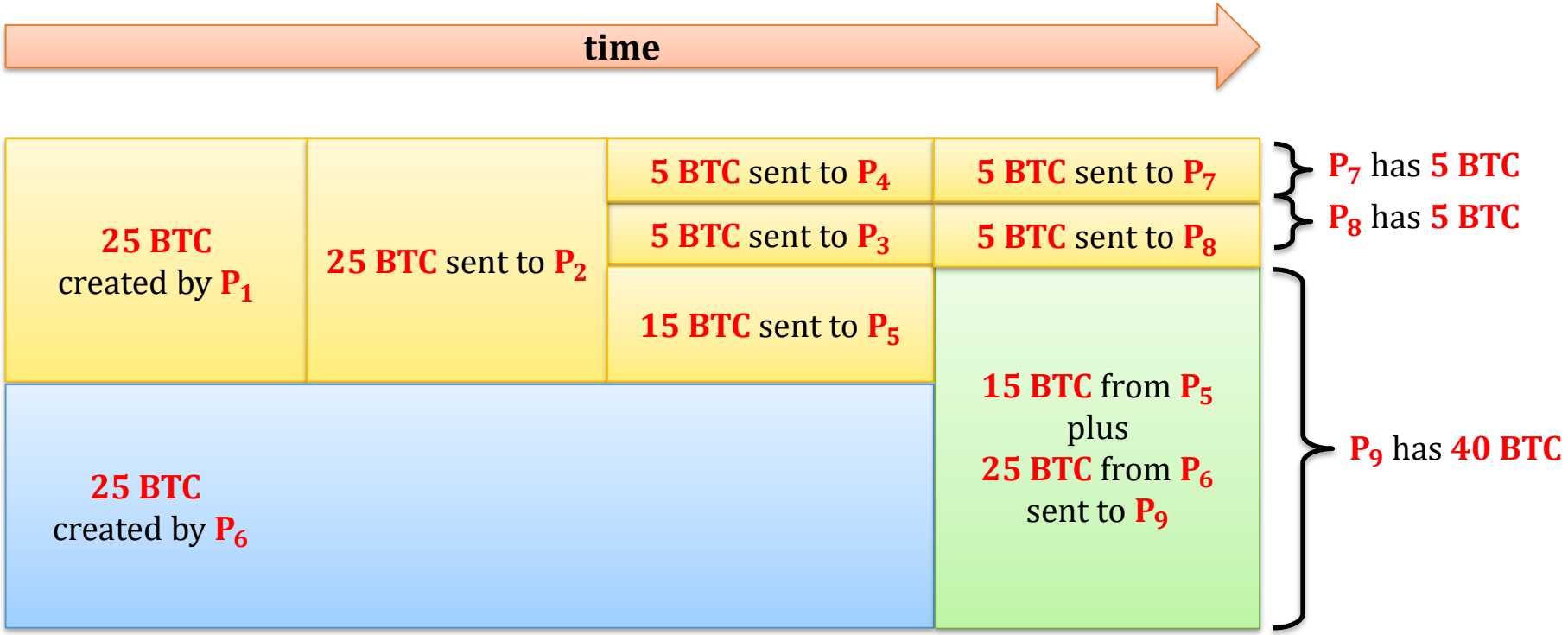
1. Introduction
2. Main design ideas of Bitcoin
    1. How is the ledger maintained?
    2. How are the users identified?
    3. Where does the money come from?
    4. What is the syntax of the transactions?

Bitcoin's money mechanics

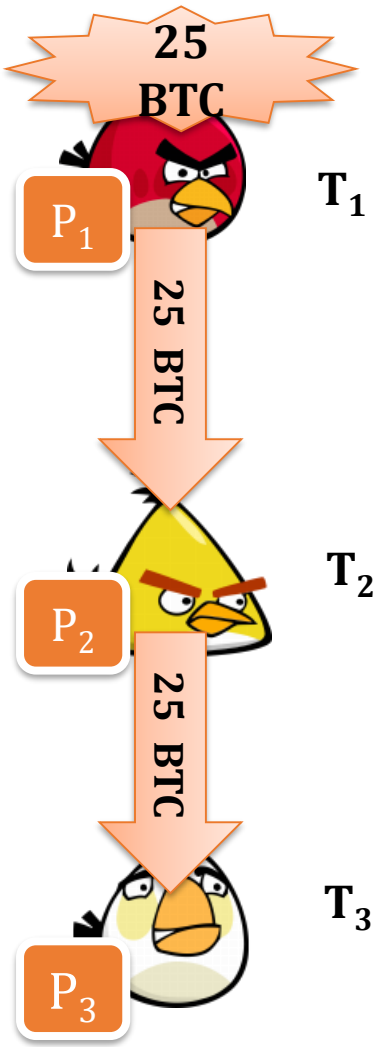
Bitcoin is "transaction based"

It uses "**UTXO** (Unspent Transaction Output) Model"

Technically: there is no notion of a "coin" in Bitcoin.



Transaction syntax – simplified view



in the "mining process"

We say that T_3 **redeems** T_2

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

$[T_2]$

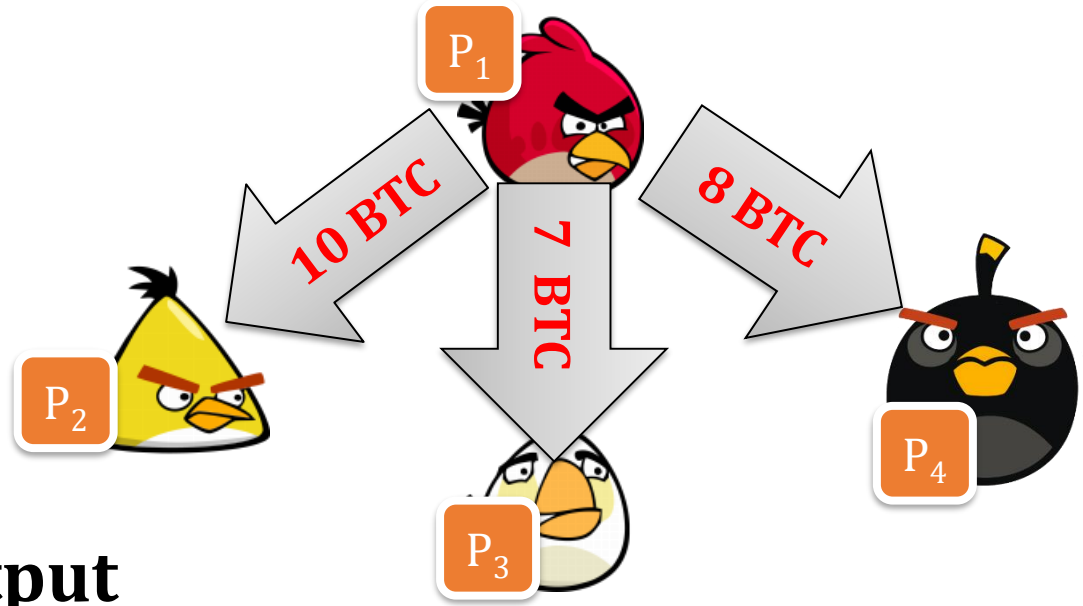
"value of T_2"

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”?

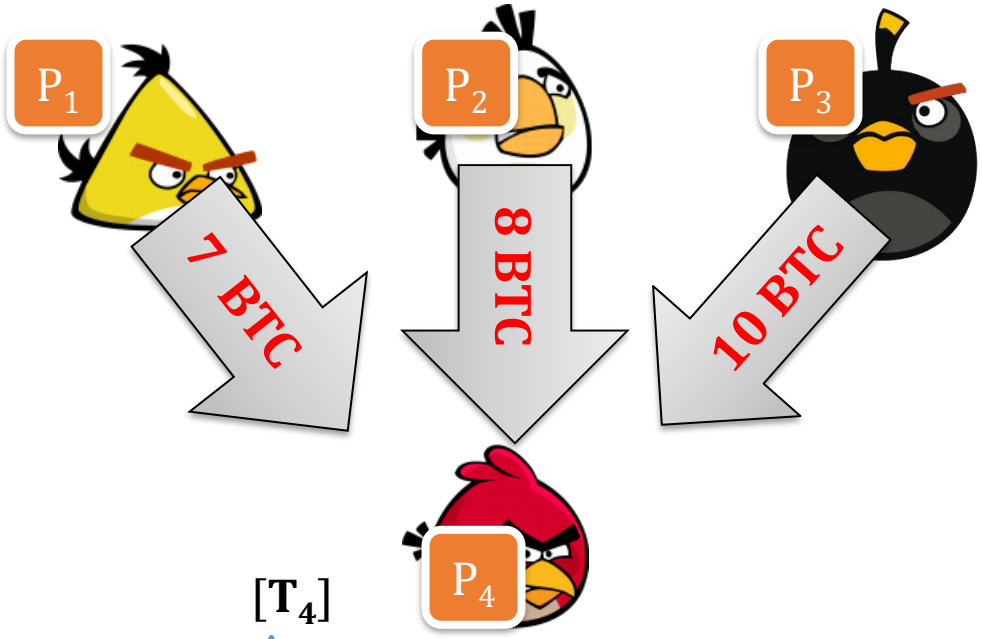


Multi-output transactions:

$[T_2]$

$T_2 =$ (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



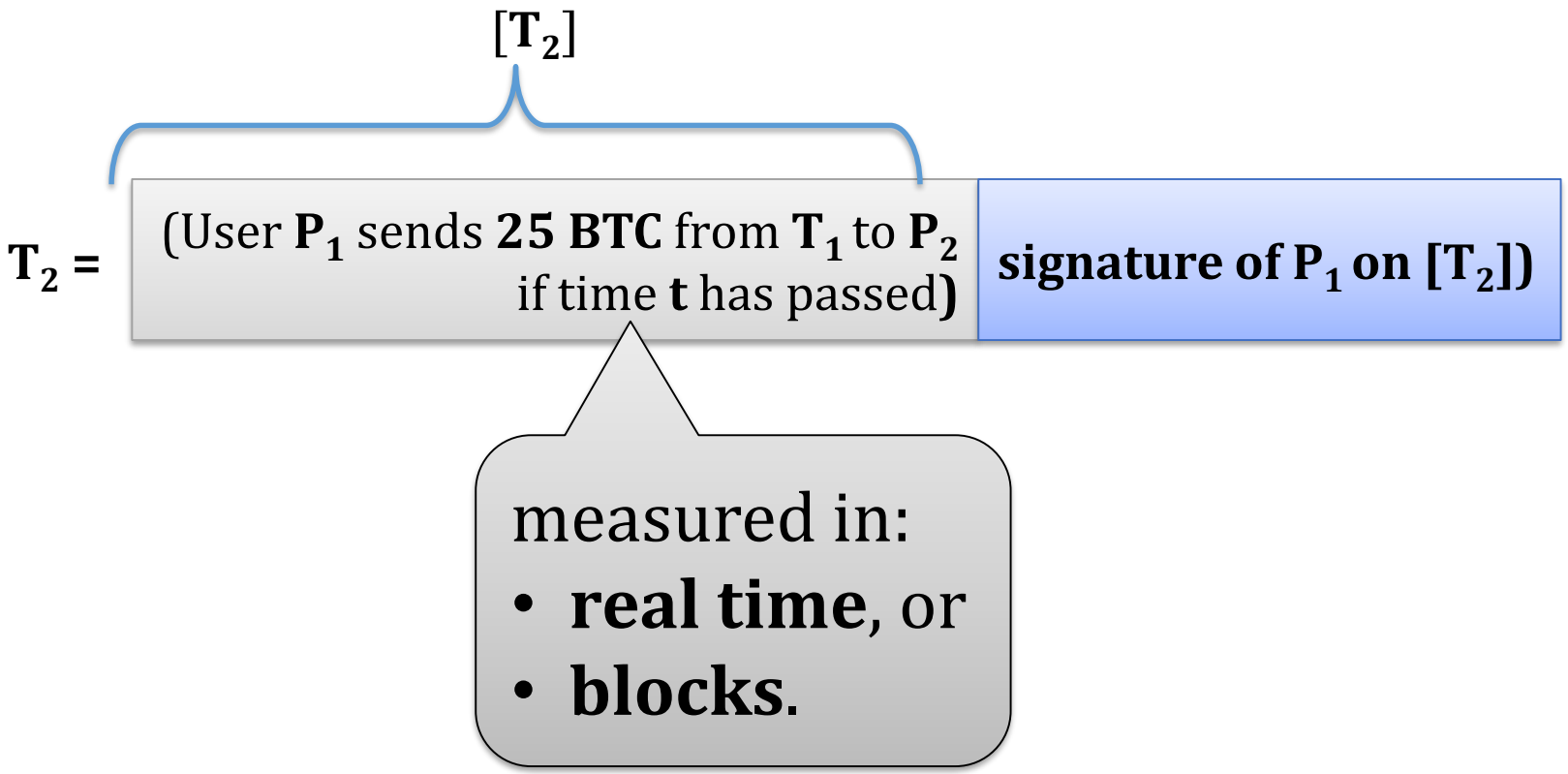
$[T_4]$

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



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 "**fee**" **and goes to the miner**)

1. The condition for redeeming a transaction can be more general (the so-called "**strange transactions**")

   we will talk about it later