CS 4770: Cryptography

CS 6750: Cryptography and Communication Security

Alina Oprea
Associate Professor, CCIS
Northeastern University

April 5 2018

Schedule

- HW 4
  - It is out on Piazza
  - Due on Thu 04/12
- Programming project 3
  - Out on 04/12
  - Due on 04/26 (last day it can be accepted)
  - Grading on 04/27
- Final exam
  - 04/23, 1-3pm

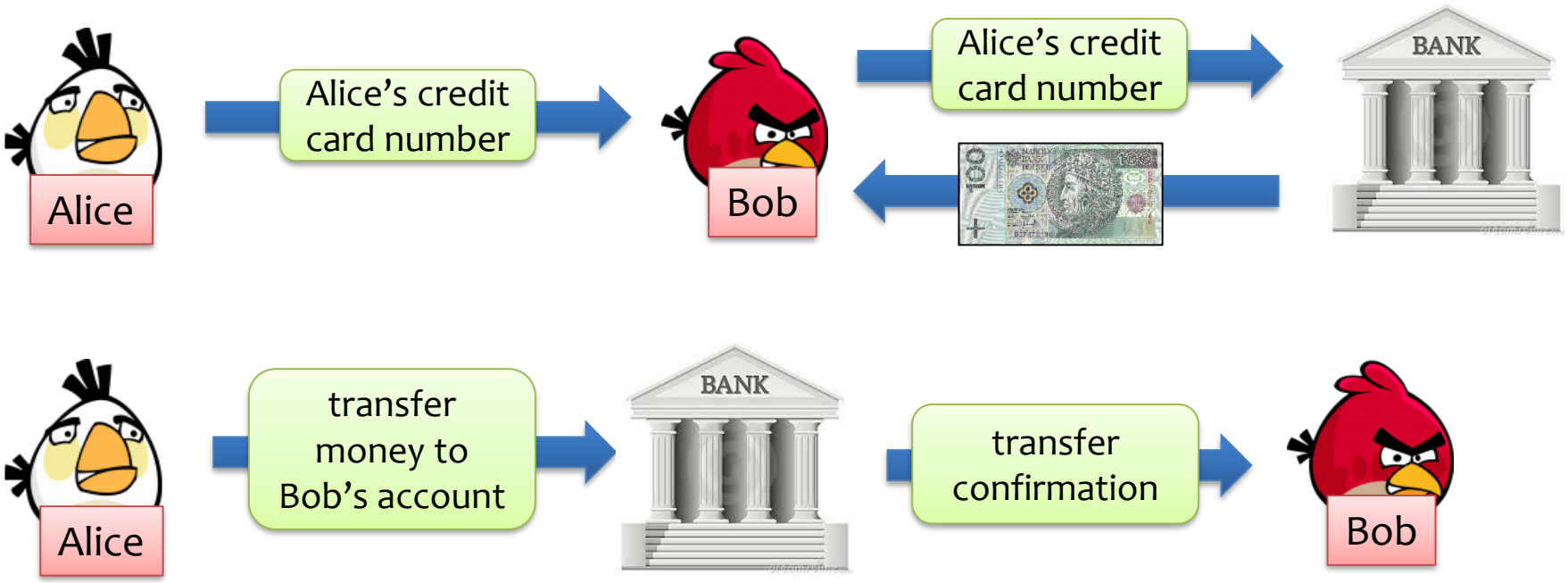
Bitcoin

- Digital crypto currencies
  - Advantages over paper cash
- Distributed public ledger
  - Blockchain creation and distribution
  - Proof of Work (PoW)
  - Agreement and resilience to adversaries
  - Incentives for users
- Bitcoin security
- Other cryptocurrencies

Resources

- Book: Bitcoin and Cryptocurrency Technologies
  - http://bitcoinbook.cs.princeton.edu/
- Bonneau et al. Research Perspectives and Challenges for Bitcoin and Cryptocurrencies.
  - https://eprint.iacr.org/2015/261.pdf
- Bitcoin and Cryptocurrency Technologies Course
  - https://www.coursera.org/learn/cryptocurrency
  - https://piazza.com/princeton/spring2015/btctech/resources

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*

Currency unit: **Bitcoin (BTC) 1 BTC = 10^8 Satoshi;** value ≈ $6800

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:

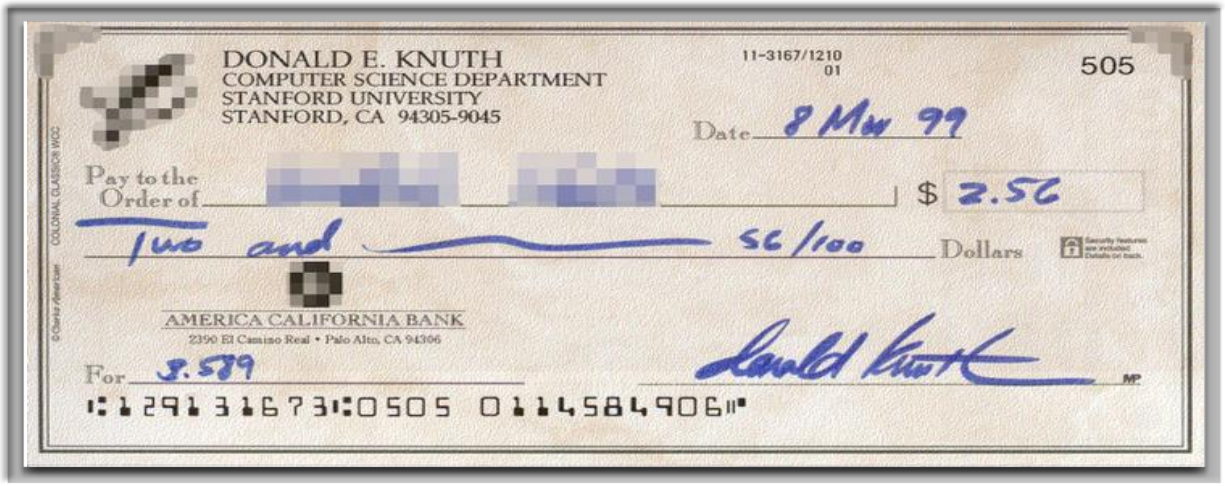
It’s a Ponzi scheme

P. Krugman

A. Greenspan

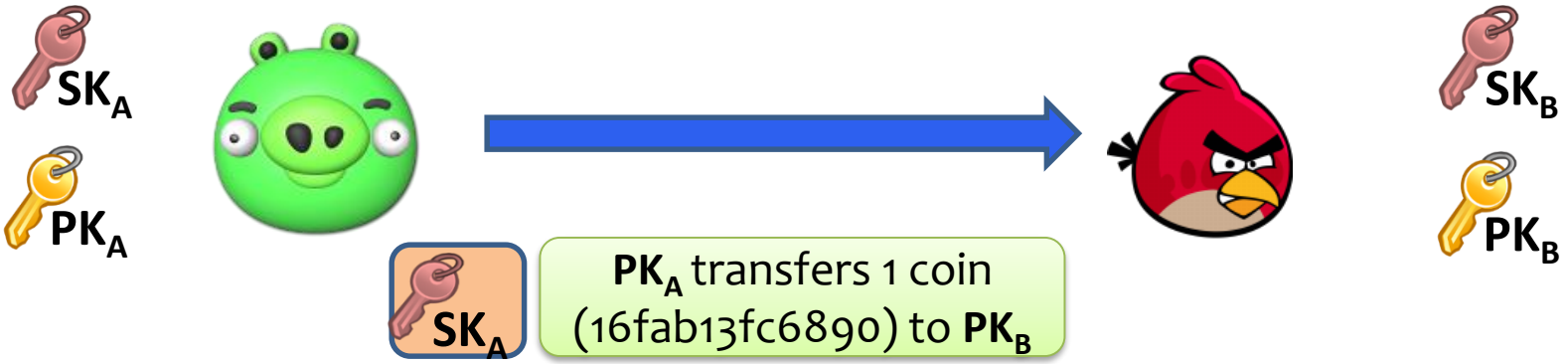
Strawman protocol

- Alice owns a coin and wants to transfer to Bob
  - Transactions can not be forged
  - 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!

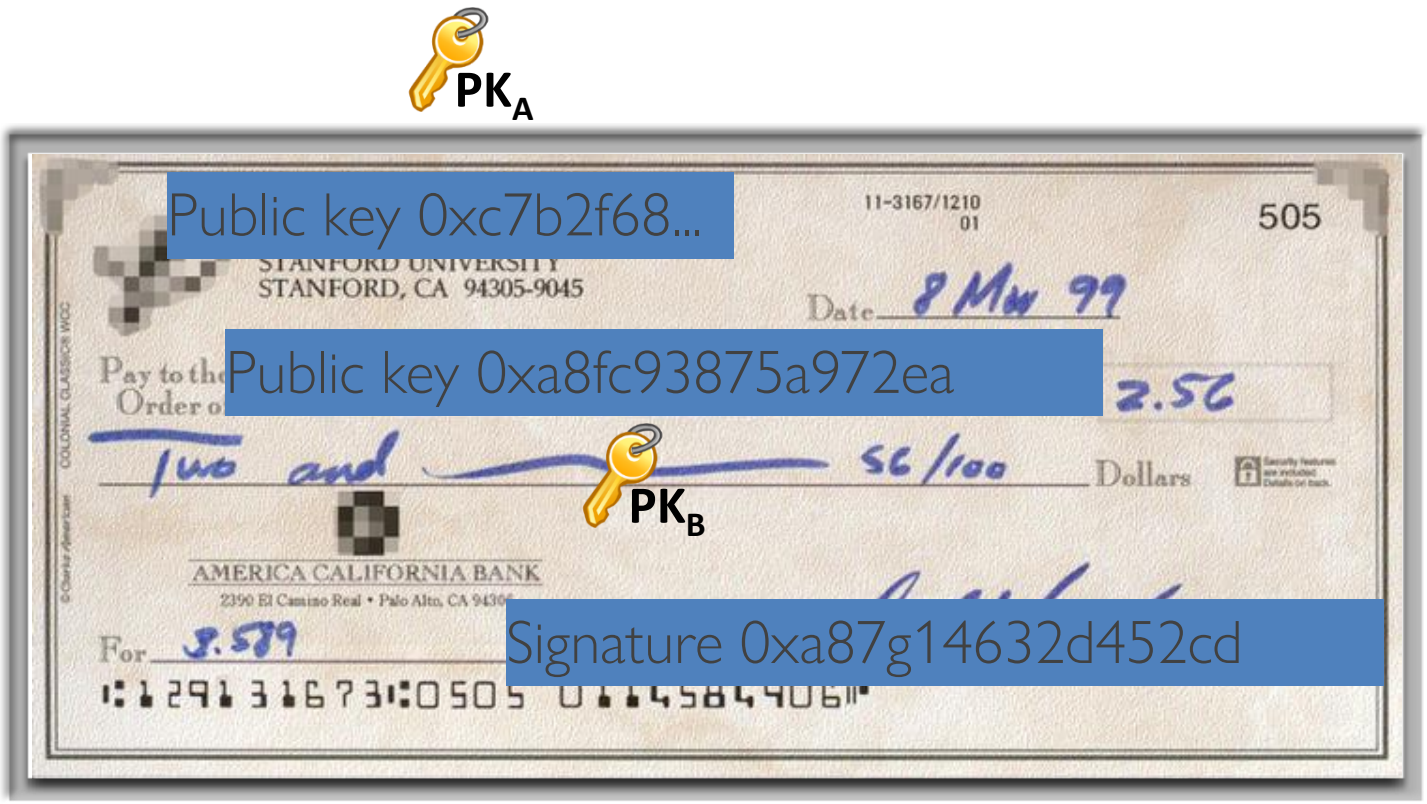


Strawman protocol

- Alice owns a coin and wants to transfer to Bob
  - Transactions can not be forged
  - 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

The Times 03/Jan/2009 *Chancellor on brink of second bailout for banks.*

Bitcoin: A Peer-to-Peer Electronic Cash System

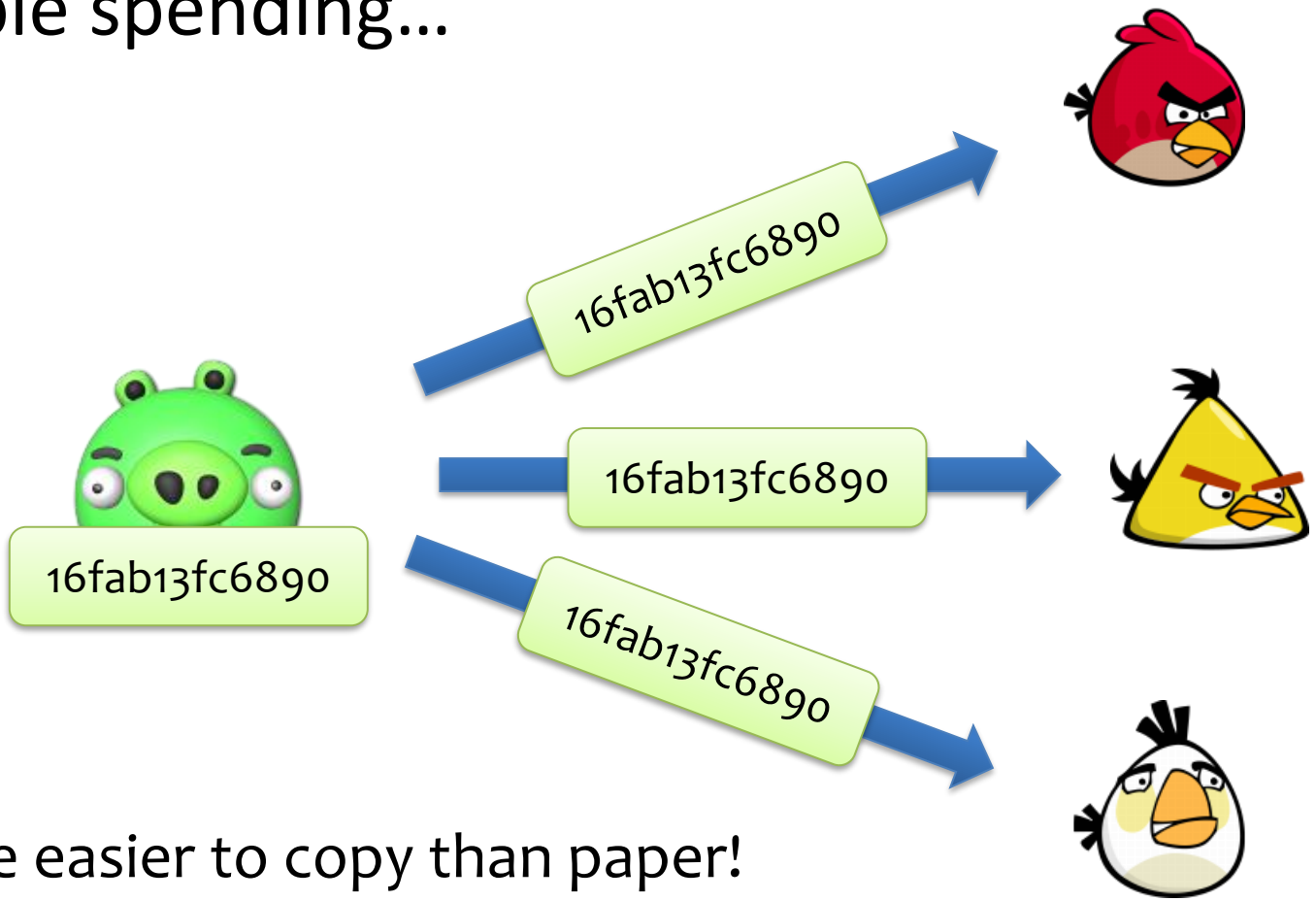
Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

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

```
bitcoin-0.1.0.rar
bitcoin-0.1.0.tgz
```

Main problem with the digital money

Double spending…

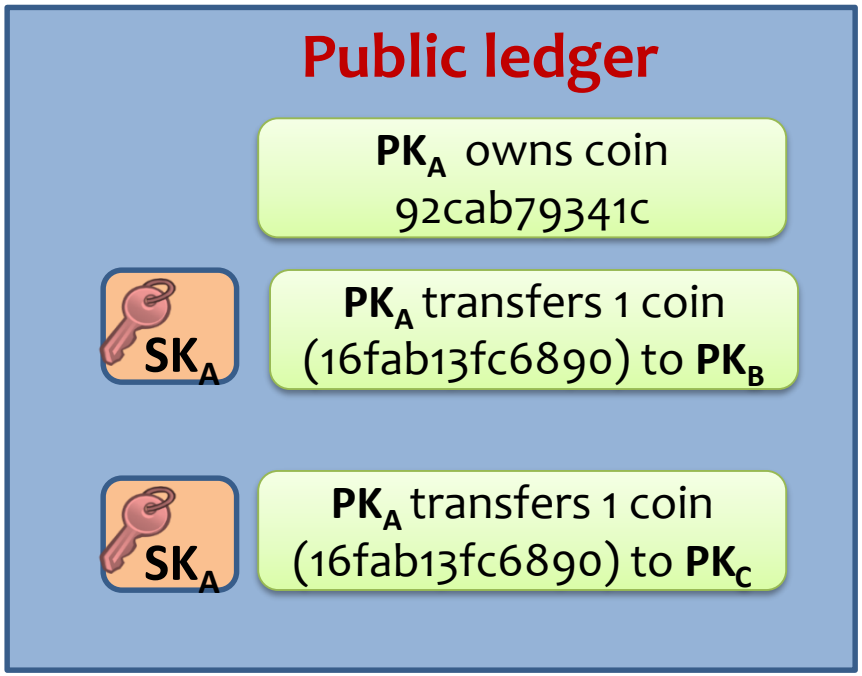


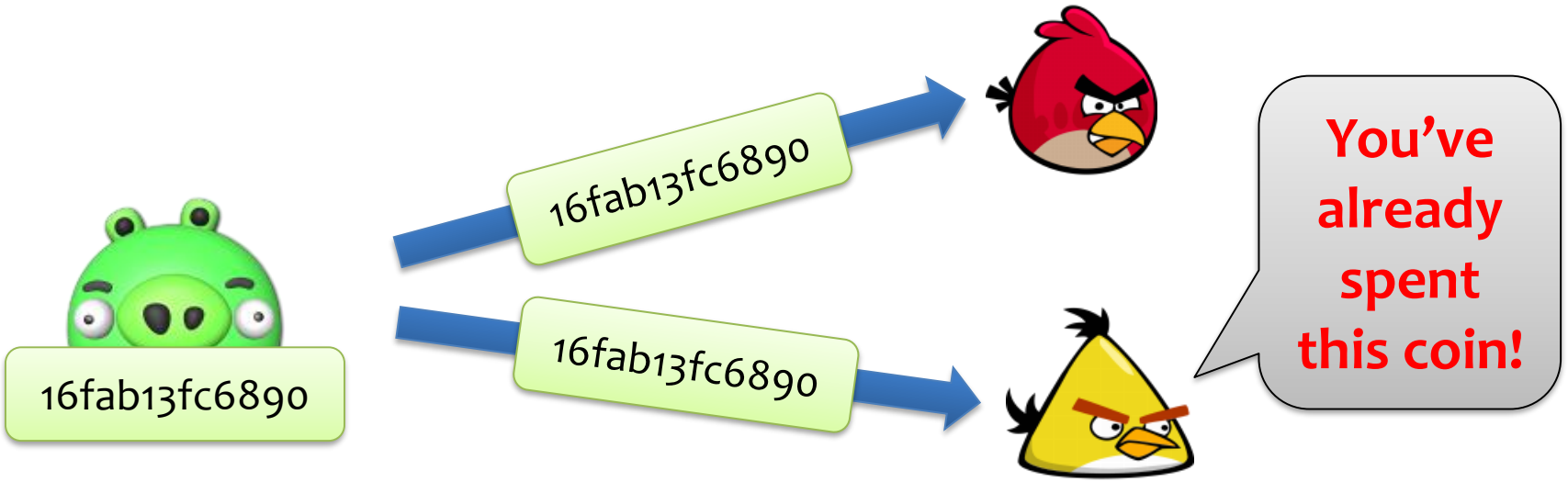
Bits are easier to copy than paper!
Signatures alone do not prevent this

Bitcoin idea

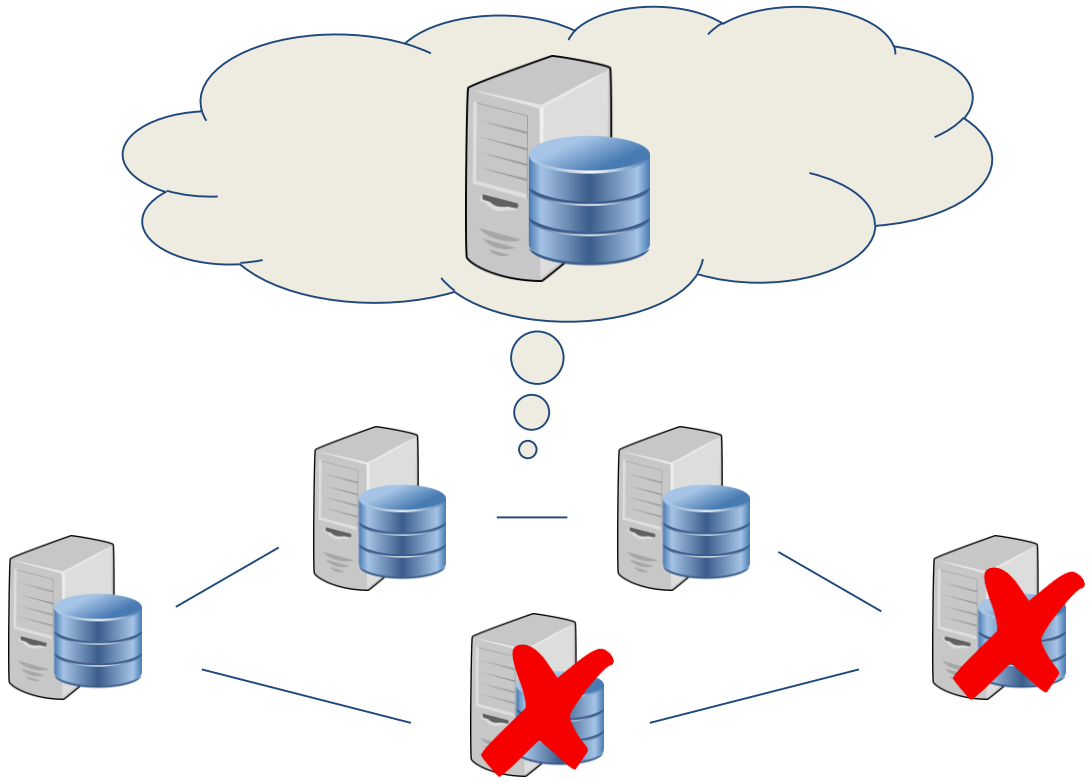
Public trusted bulletin-board (public ledger or DB)

- Includes list of all transactions
- Verifiable by all users
- How to maintain it in decentralized fashion?





A blockchain is a *Distributed System*



P2P Network

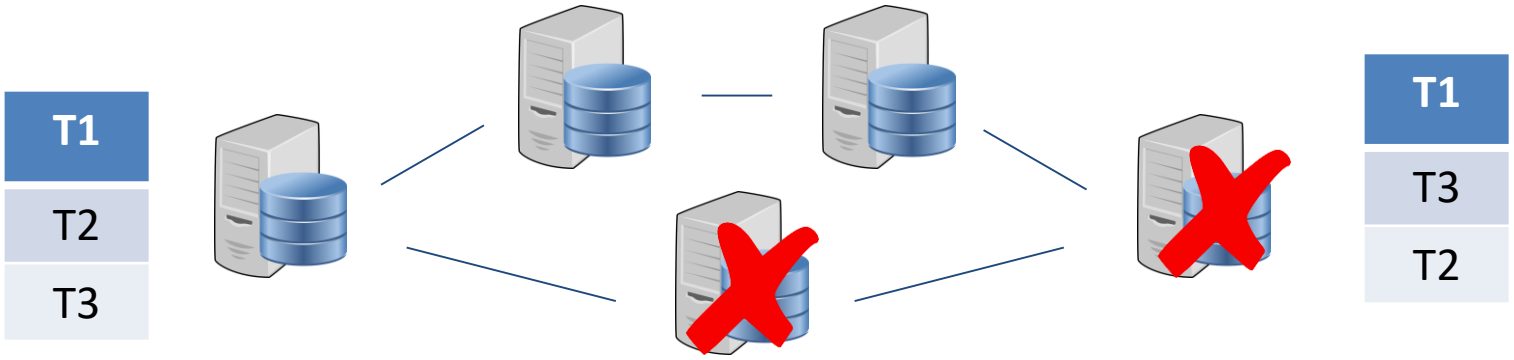
Ordinary databases:

- Distributed within one domain
- For performance and availability

Decentralized Ledger:

- Distributed across multiple entities
- Privacy and security against attacks
- Correctness assuming honest majority
- No single point of failure

Challenges in designing public ledger



- **Decentralized ledger**
  - Each user maintains a list of transactions ordered across time
  - His own transactions and transactions received from other users
- **Main challenges: Obtain consensus**
  - Order of transactions is the same at all nodes
- **Attack models**
  - Network failures (messages might not be delivered timely)
  - Offline participants (nodes leave and re-connect to the network)
  - Malicious nodes (nodes try to double spend)

Key insights

- Decentralization through P2P network
  - Each transaction is *broadcast* to all nodes
  - Each node keeps a tamper-evident log (local ledger) of **all** Bitcoin transactions
  - Nodes agree on list of transactions and their order (distributed consensus)
- Consensus happens over longer periods of time
  - Probabilistic guarantees
  - In online transactions can have some delay
- Attack model
  - Assumption: attacker cannot control majority computational power in the network

Bitcoin

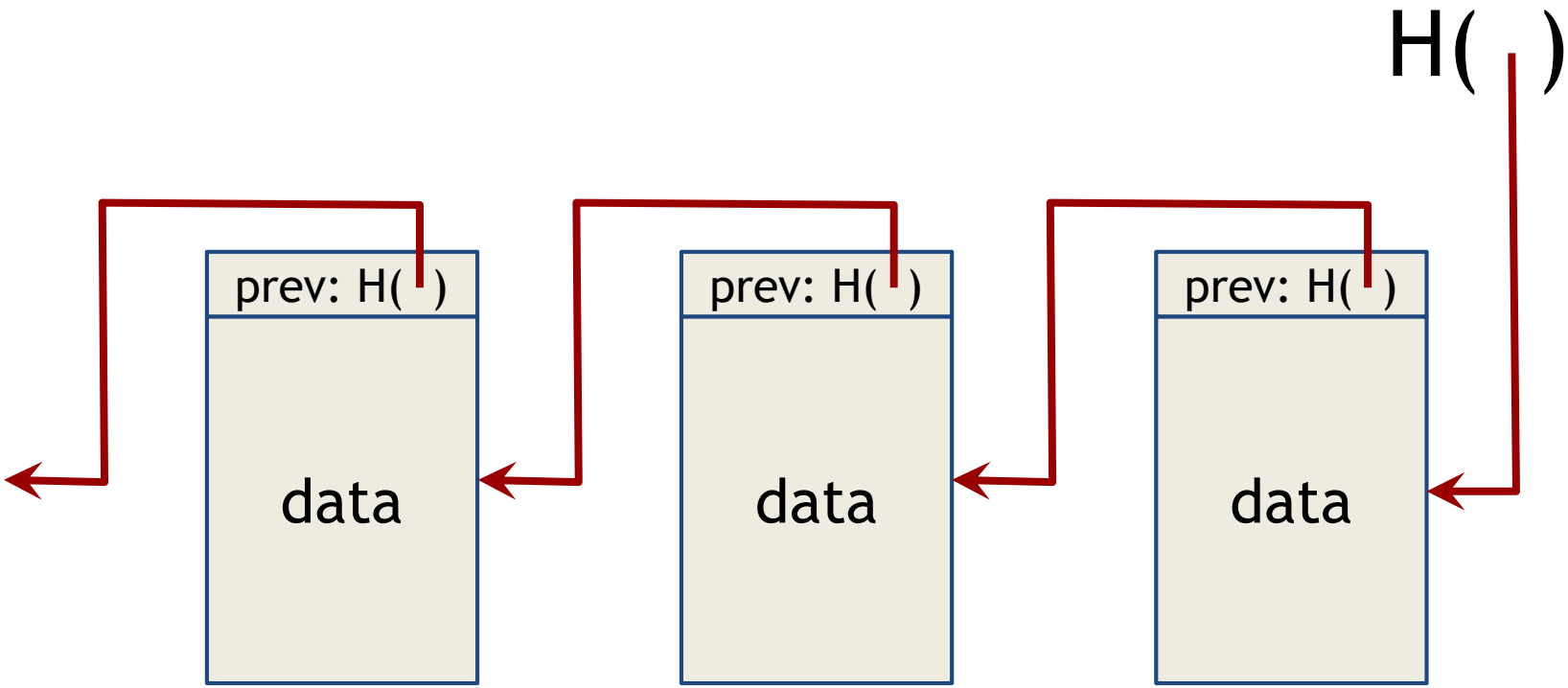
- Digital crypto currencies
  - Advantages over paper cash
- Distributed public ledger
  - Blockchain creation and distribution
  - Proof of Work (PoW)
  - Agreement and resilience to adversaries
  - Incentives for users
- Bitcoin security
- Other cryptocurrencies

Public ledger

- Tamper-evident log
  - Record and order all transactions locally at each node
  - Valid transactions can not be modified
  - New transactions are appended after being validated
- How to design it?
  - What data structure and crypto primitives to use?
- How to prevent attackers controlling majority of transactions?
- How to incentivize users?
- How to reach agreement?

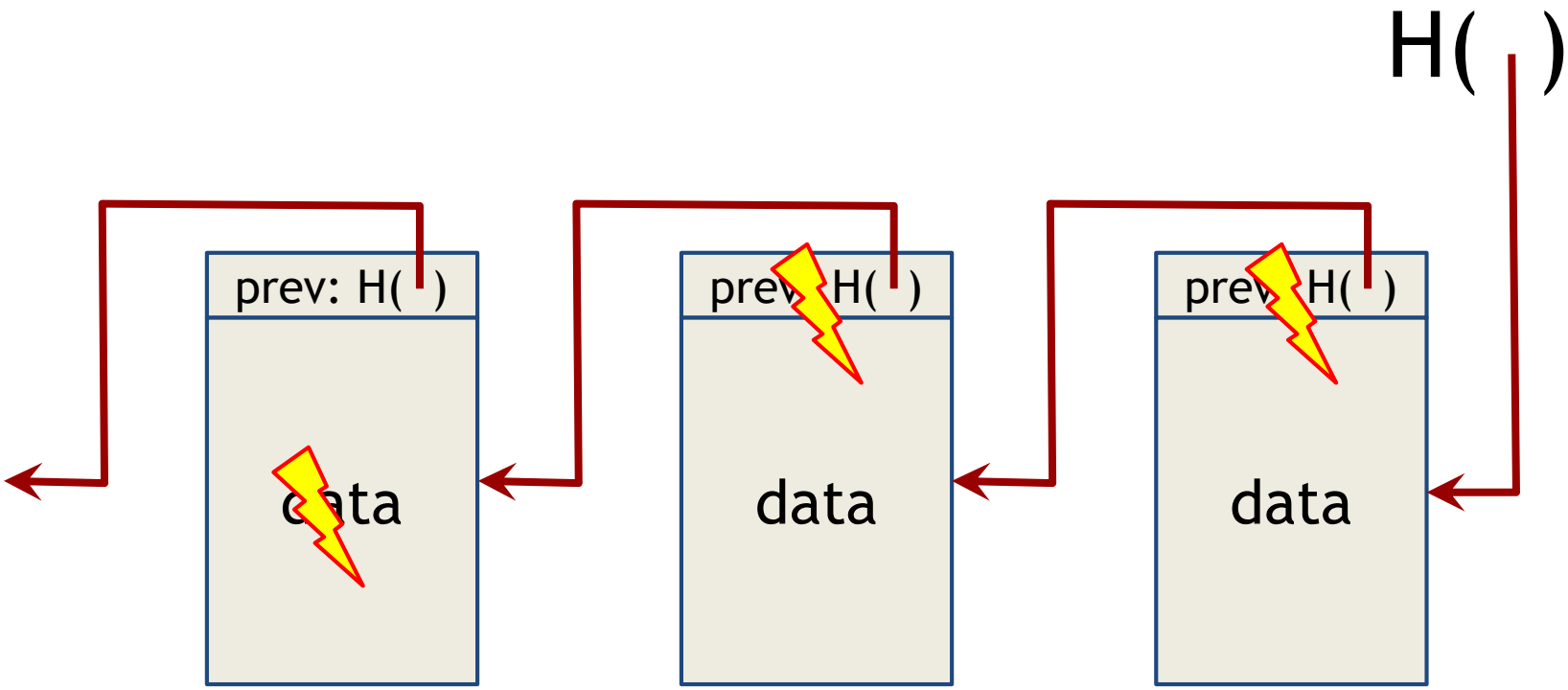
Block chain

Linked list with hash pointers



Tamper-evident log

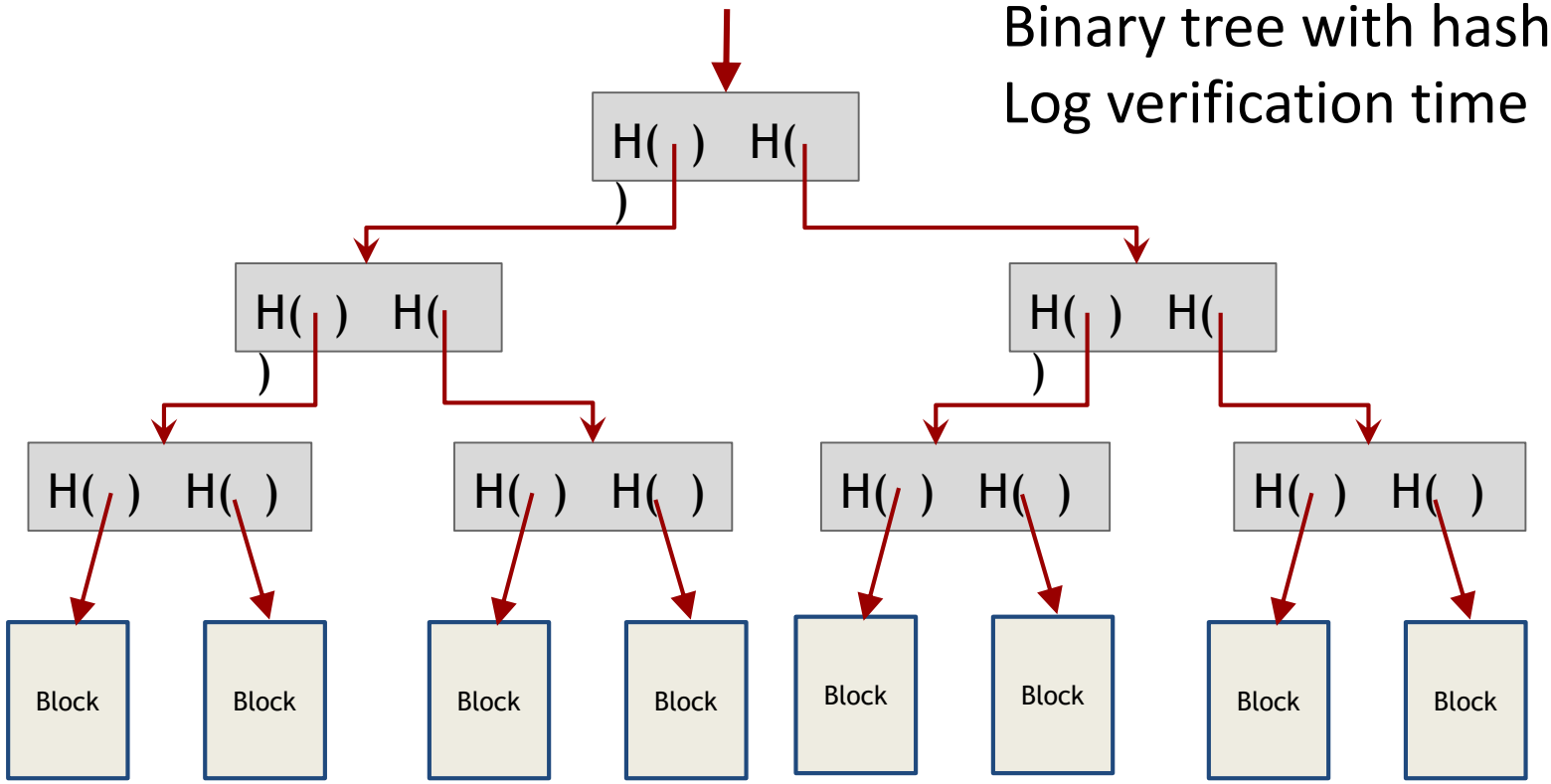
Detecting tampering



Tamper-evident log

Merkle trees

Binary tree with hash pointers
Log verification time



Block

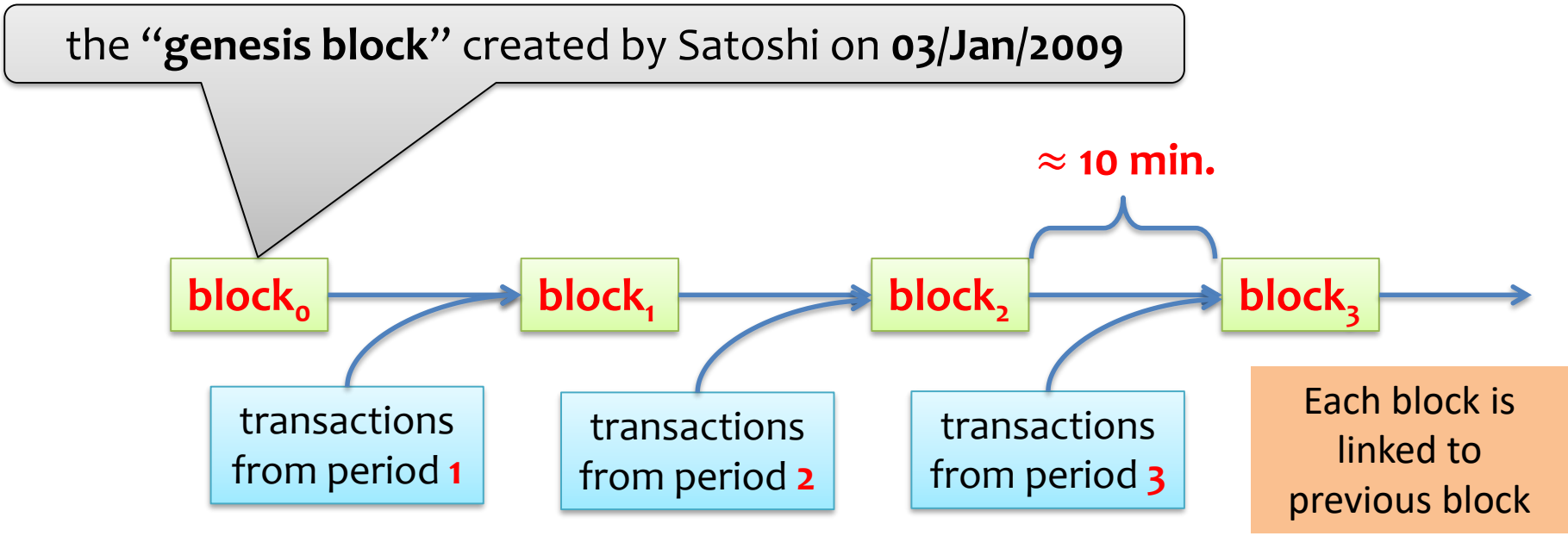
Block Hash	Prev. Block Hash	Nonce
Transaction *I*		
Transaction *J*		
Transaction *K*		

Multiple transactions for efficiency

Block chain

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

They maintain a chain of blocks (blockchain):



Distributing transactions

- New transactions are *broadcast* to all nodes
  - P2P network
- Each node *collects new transactions* into a block
- In each round (e.g., every 10 minutes)
  - A random node *creates the next block* (includes outstanding transactions)
- Other nodes accept the block only if *all transactions in it are valid*
  - Valid signatures
  - Coins not spent before
- Nodes accept the block by including it in their local ledger

Public ledger

- Tamper-evident log
  - Record and order all transactions
  - Valid transactions can not be modified
  - New transactions are appended after being validated
- How to design it?
  - What data structure and crypto primitives to use?
- How to prevent attackers controlling majority of transactions?
- How to incentivize users?
- How to reach agreement?

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

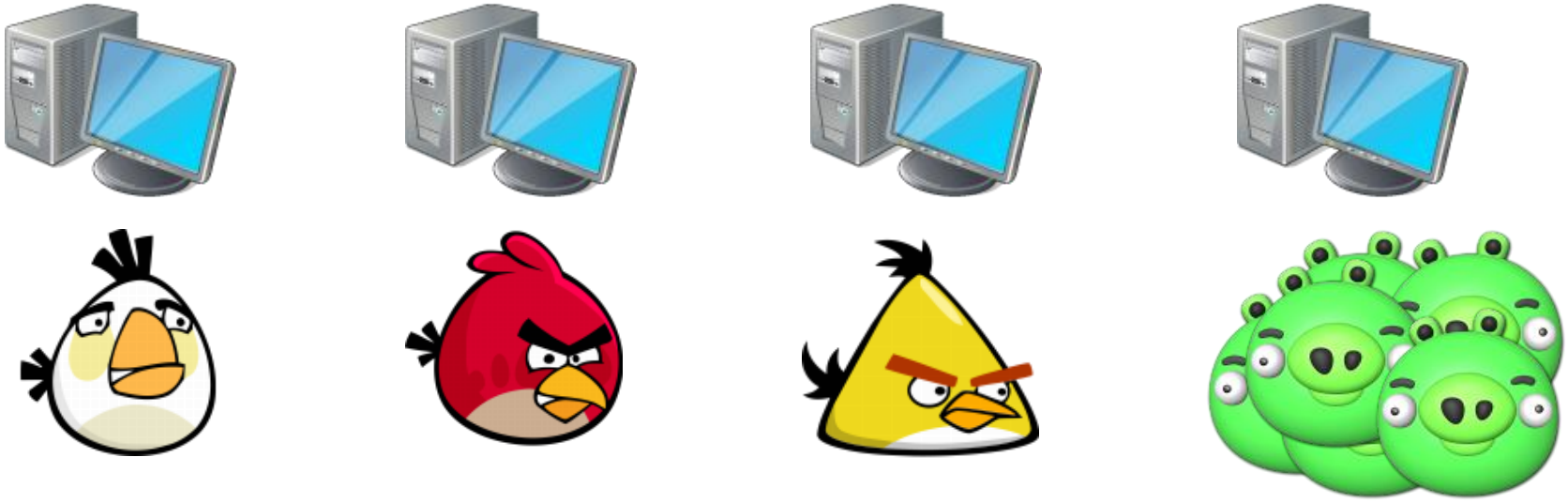
Use a resource that is hard to obtain

- 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

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)

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 (ASIC)**
- 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)

s

Verifier

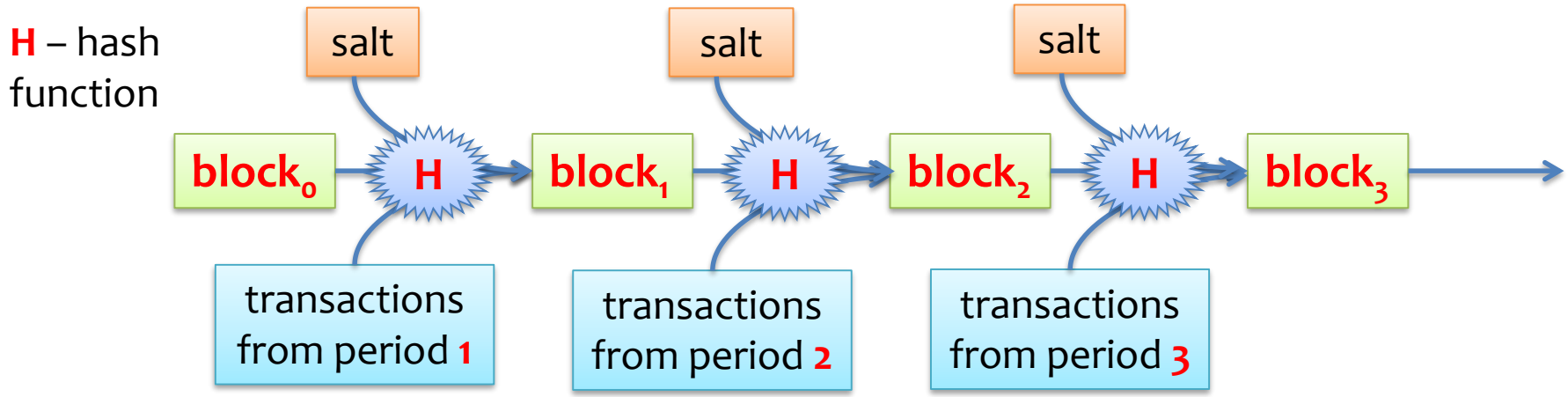
checks if **H(s,x)** starts with **n** zeros

salt	hardness parameter n

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

takes time **TIME(H)**

How are the PoWs used?



Main idea: to extend it one needs to find **salt** such that

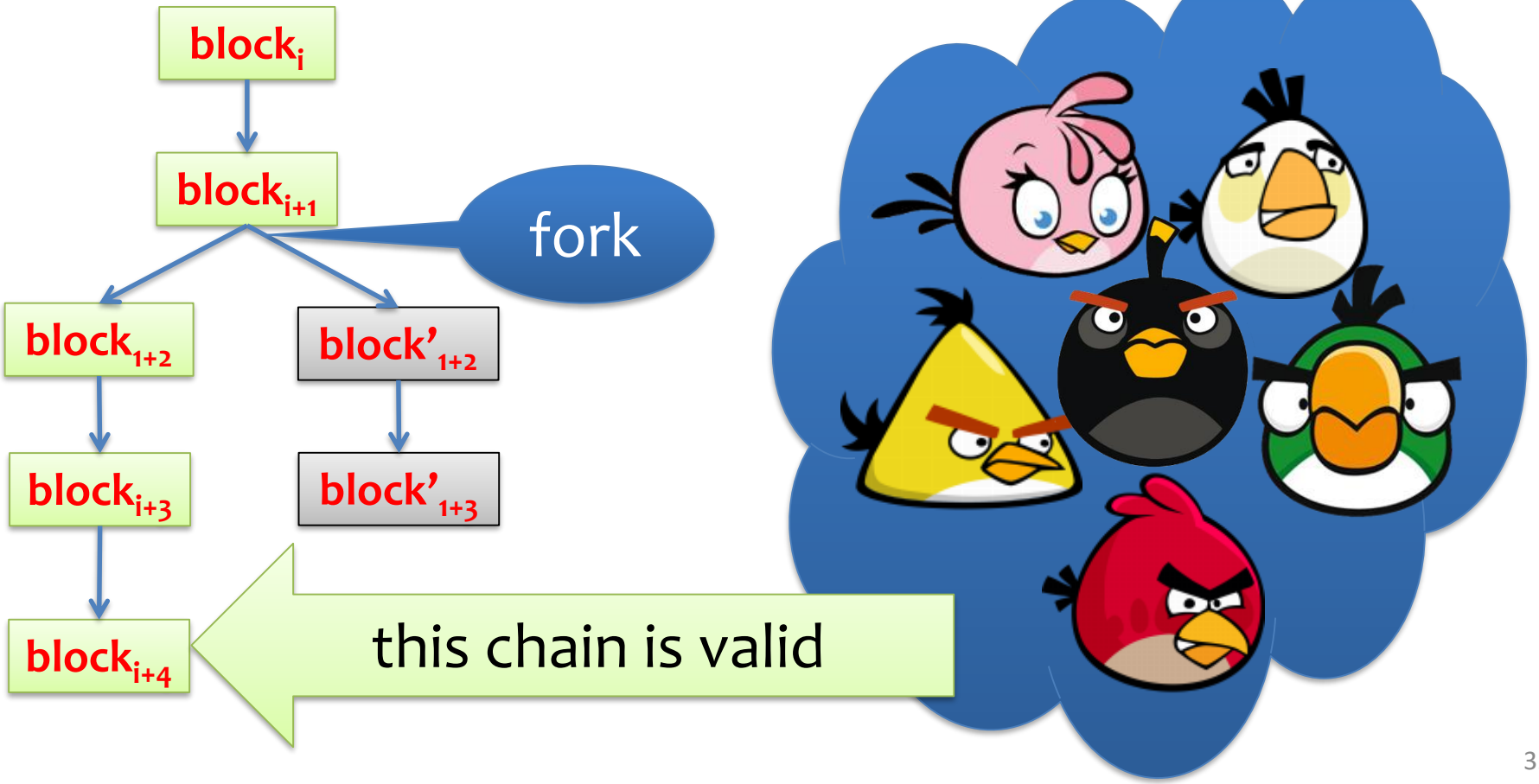
H(salt, block$_i$, transactions) starts with some number **n** of **zeros**

Process is called block mining

Double spending: “forks”

The “**longest**” chain counts.

- It includes “more work”



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/