Proof-of-Work Blockchains and Smart Contract Security

Martin Derka, Ph.D.
Senior Research Engineer

Quantstamp™

Talk structure

1. **Consensus and Proof-of-Work**
2. **Blockchain as a programmable platform**
3. **Possible security flaws**
4. **Research topics**

Satoshi Nakamoto (2008)
Bitcoin: A Peer-to-Peer Electronic Cash System

Vitalik Buterin (2013)
Ethereum: The Ultimate Smart Contract and Decentralized Application Platform

Consensus and Proof-of-Work

Consensus replacing banks

Imagine that you have money at a bank...

Consensus replacing banks

Imagine that you have money at a bank...

You trust that the bank:

1. Handles the balances honestly
2. Does not prevent you from access
3. Does not make errors
4. Does not go out of business
5. Protects your personal data
6. Employs honest employees
7. ...

Consensus replacing banks

Is there another way of owning assets?

Consensus:
General agreement. Everybody knows and agrees that you own assets.

Developing a consensus

Set up:

1. Students gather in a classroom.
2. Everyone maintains a personal copy of the classroom's ledger.
3. Proposed transactions are written on a board (in parallel).

In order to achieve consensus, we all need to execute the same transactions in our copies of the ledger.

Developing a consensus

Set up:

1. Students gather in a classroom.
2. Everyone maintains a personal copy of the classroom's ledger.
3. Proposed transactions are written on a board (in parallel).

In order to achieve consensus, we all need to execute the same transactions in our copies of the ledger.

Who chooses which transactions will be executed and in which order?

What can go wrong?

Martin's original balance: $1000

Transactions

$700 from Martin to Alice
$700 from Martin to Bob

What can go wrong?

Martin's original balance: $1000

Transactions

$700 from Martin to Alice
$700 from Martin to Bob

Ledger 1

```
1.  Martin  $1000
2.  Martin  -$700
3.  Alice   +$700
4.  Reject Martin to Bob
```

What can go wrong?

Martin's original balance: $1000

Transactions

$700 from Martin to Alice
$700 from Martin to Bob

Ledger 1

```
1.  Martin   $1000
2.  Martin   -$700
3.  Alice    +$700
4.  Reject Martin to Bob
```

Ledger 2

```
1.  Martin   $1000
2.  Martin   -$700
3.  Bob      +$700
4.  Reject Martin to Alice
    ...
```

Need to have a leader

We still need a leader who will determine the truth.
(such as the order of transactions)

One leader can be unfair.
(This is the bank)

We want a different leader every single time.

Problems with marathons

Choose leader by a marathon race.

Problems with marathons

Choose leader by a marathon race.

Problems:

1. Authenticity of transactions
2. Privacy of transactions
3. Co-location of participants and transaction transmission
4. Learning the history of the ledger and trusting it
5. It will be very slow because organizing and running marathons takes time
6. What is the motivation to run marathons for others?

Digital Proof-of-Work

Marathon race is analogous to what we call a **Proof-of-Work:**

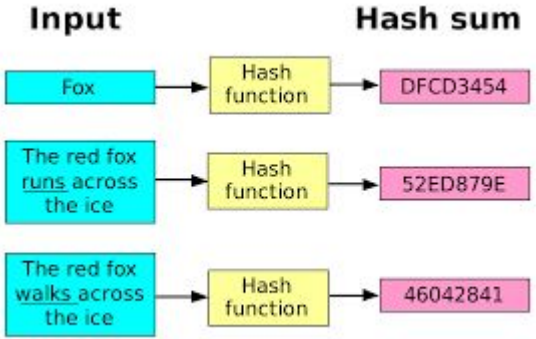
You do something that takes lots of effort. If you do it faster than anyone else, you become the leader.

In the digital world, the equivalent of **work** is **computing**.

Proof-of-Work: Spending computing power to solve some puzzle.

Hashing

Hash function is a function that maps data of an arbitrary length to a short fixed-length representation.

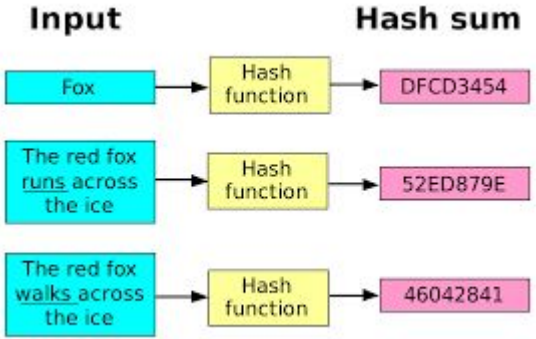


Hashing

Hash function is a function that maps data of an arbitrary length to a short fixed-length representation.

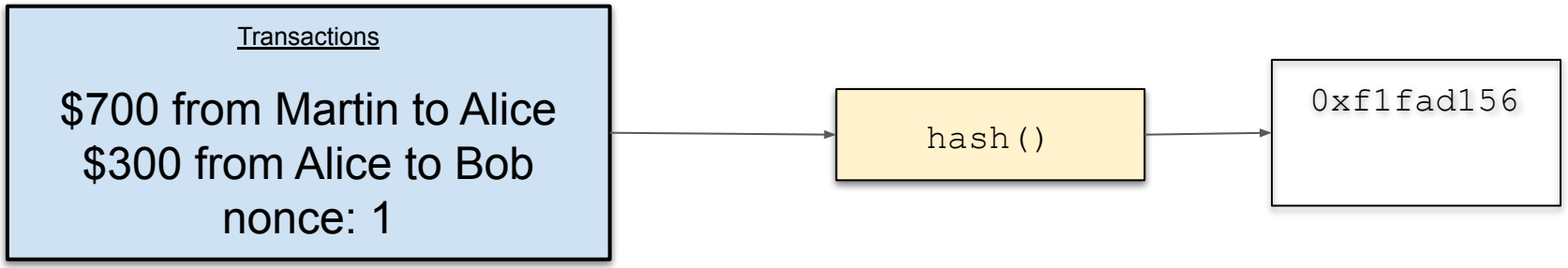
Good hash functions are:

1. Irreversible
2. Unpredictable
3. Evenly distributed
4. Other colliding inputs cannot be guessed



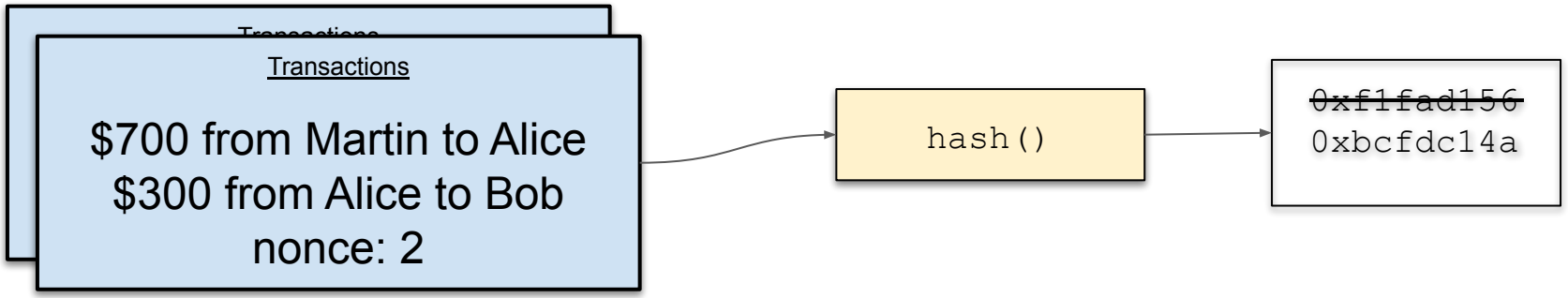
Proof-of-Work puzzle

1. Take transactions that you want to append to the ledger.
2. Append a number (nonce) to them and hash the message
3. Find a nonce such that the produced hash starts with “00” (or with another restricted prefix)



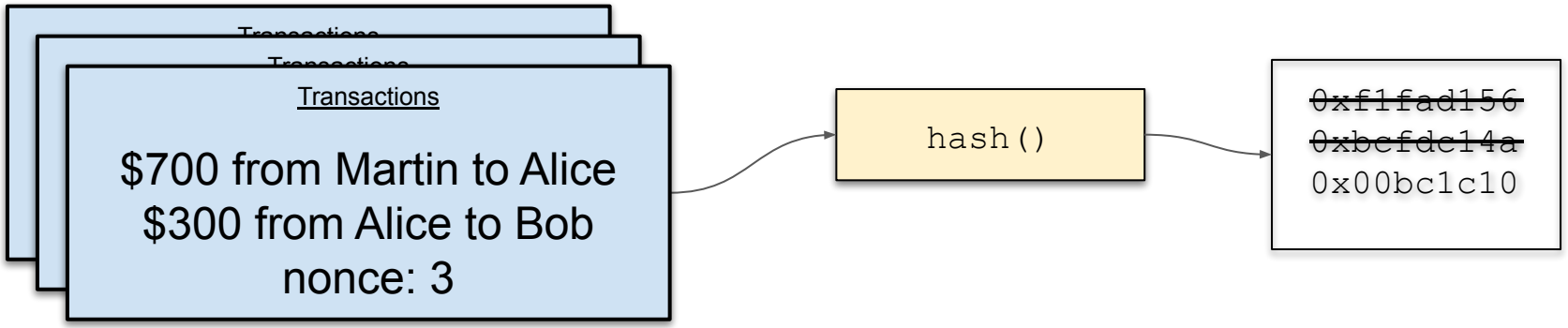
Proof-of-Work puzzle

1. Take transactions that you want to append to the ledger.
2. Append a number (nonce) to them and hash the message
3. Find a nonce such that the produced hash starts with “00” (or with another restricted prefix)

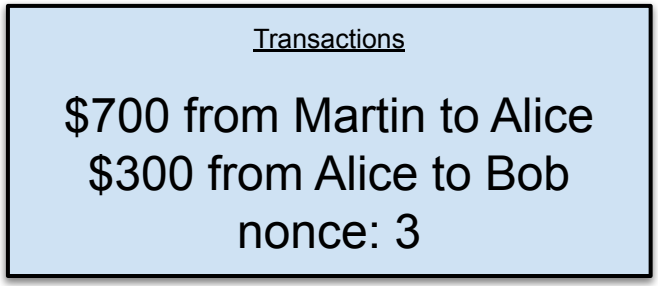


Proof-of-Work puzzle

1. Take transactions that you want to append to the ledger.
2. Append a number (nonce) to them and hash the message
3. Find a nonce such that the produced hash starts with “00” (or with another restricted prefix)



Linking the blocks



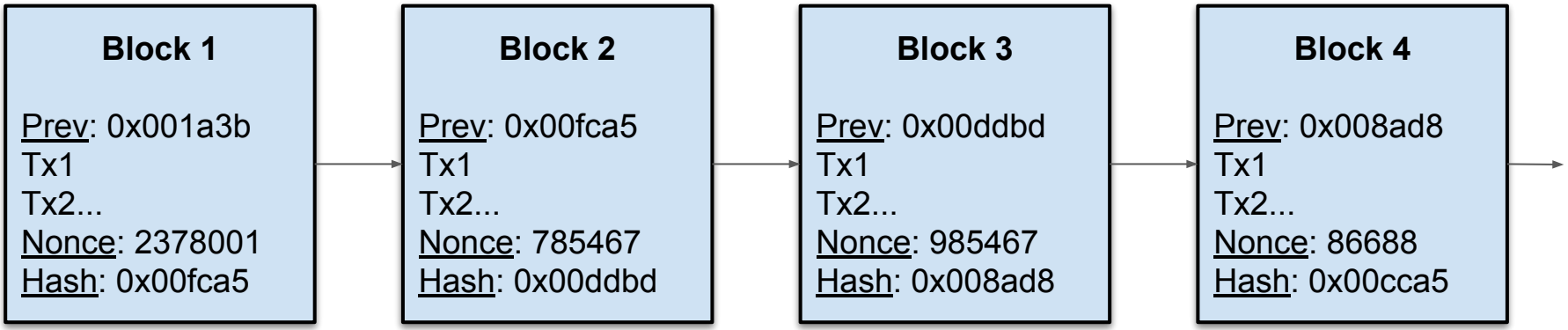
= Block (almost)

Linking the blocks

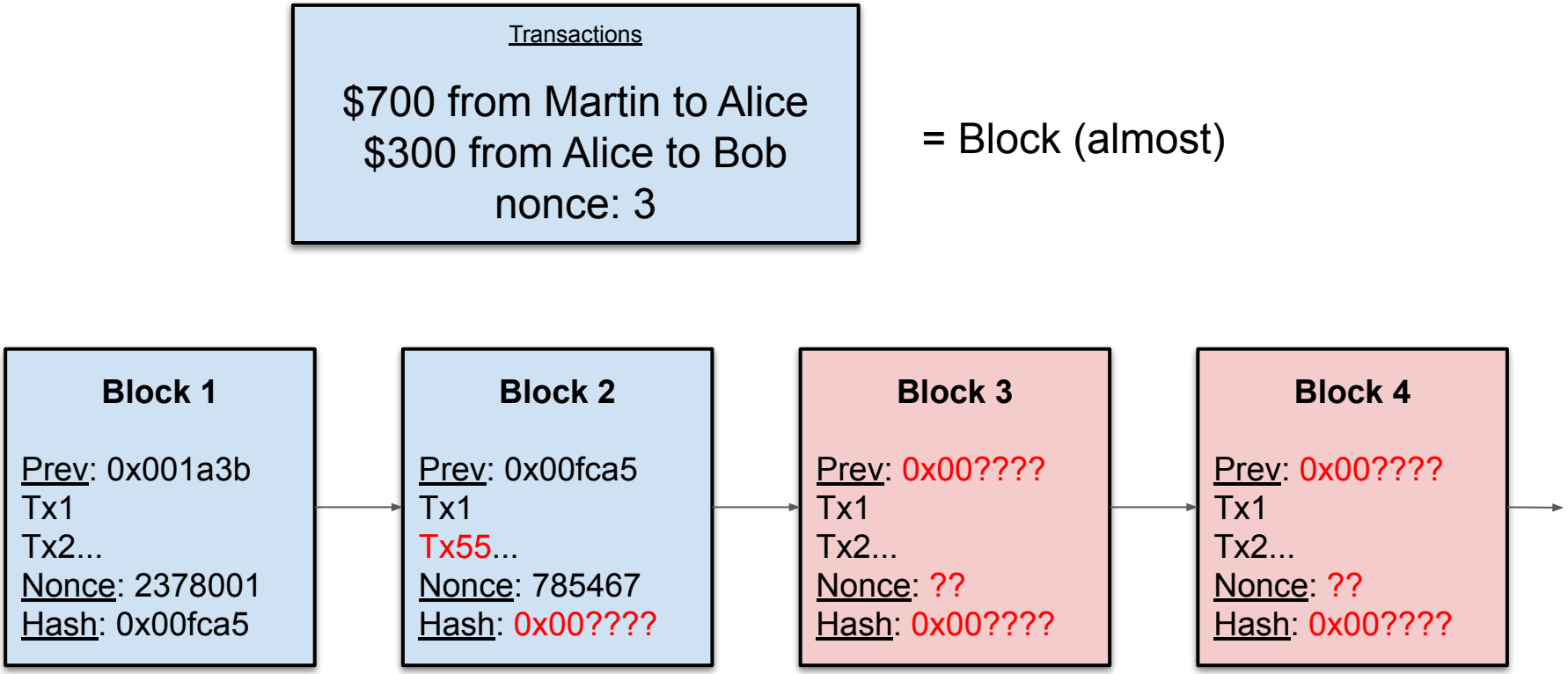
Transactions

$700 from Martin to Alice
$300 from Alice to Bob
nonce: 3

= Block (almost)



Linking the blocks



Proof-of-Work blockchain

1. Transactions and blocks are broadcasted over a P2P network.

Proof-of-Work blockchain

1. Transactions and blocks are broadcasted over a P2P network.
2. Every participant maintains their own chain of blocks, ledger, and candidate transaction pool.

Proof-of-Work blockchain

1. Transactions and blocks are broadcasted over a P2P network.
2. Every participant maintains their own chain of blocks, ledger, and candidate transaction pool.
3. Every participant can select transactions for the next block and start calculating the proof of work for:
    i. Hash of the previous block
    ii. All transactions inside of this block
    iii. An integer (so-called nonce) that will produce hash smaller than some constant (i.e., starting with 0x00...0)

Proof-of-Work blockchain

1. Transactions and blocks are broadcasted over a P2P network.
2. Every participant maintains their own chain of blocks, ledger, and candidate transaction pool.
3. Every participant can select transactions for the next block and start calculating the proof of work for:
    i. Hash of the previous block
    ii. All transactions inside of this block
    iii. An integer (so-called nonce) that will produce hash smaller than some constant (i.e., starting with 0x00...0)
4. The first participant to succeed broadcasts the block to the network.

Proof-of-Work blockchain

1. Transactions and blocks are broadcasted over a P2P network.
2. Every participant maintains their own chain of blocks, ledger, and candidate transaction pool.
3. Every participant can select transactions for the next block and start calculating the proof of work for:
    i. Hash of the previous block
    ii. All transactions inside of this block
    iii. An integer (so-called nonce) that will produce hash smaller than some constant (i.e., starting with 0x00...0)
4. The first participant to succeed broadcasts the block to the network.
5. All participants prune the transactions and repeat.

Proof-of-Work using hash functions

Nonce = An integer that complements the block for it to hash to a value smaller than some constant (given by the currently set **difficulty**).

Proof-of-Work = Guessing the right nonce.

Proof-of-Work is hard to come up with, but quick to verify!

Proof-of-Work using hash functions

Nonce = An integer that complements the block for it to hash to a value smaller than some constant (given by the currently set **difficulty**).

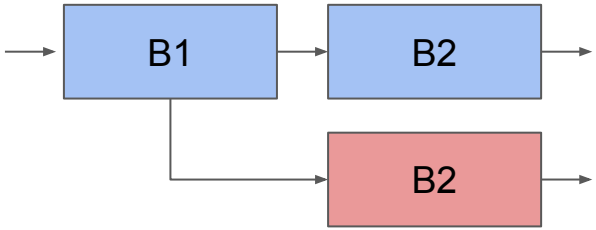
Proof-of-Work = Guessing the right nonce.

Proof-of-Work is hard to come up with, but quick to verify!

Providing the Proof-of-Work for transactions is also called **mining** (Bitcoin, Ethereum). The winner can append a special transaction that will award him with a **block reward** (“money” printed from the thin air).

Forks in chains

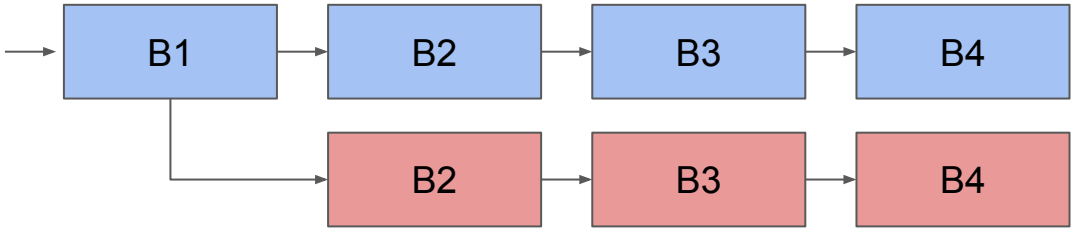
What happens if two miners arrive to a difference proof of work?



Forks in chains

What happens if two miners arrive to a difference proof of work?

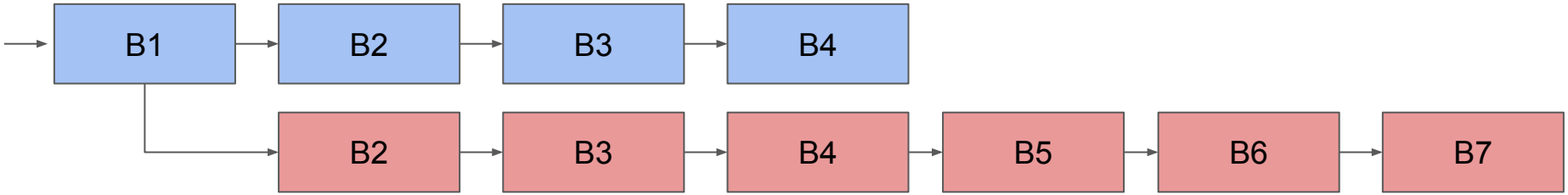
Basic assumption: Proof of work is hard. Nobody can be the leader forever.



Forks in chains

What happens if two miners arrive to a difference proof of work?

Basic assumption: Proof of work is hard. Nobody can be the leader forever.



Solution: Always trust the longest chain.

Problems with our system

Choose leader by Proof-of-Work in a P2P network.

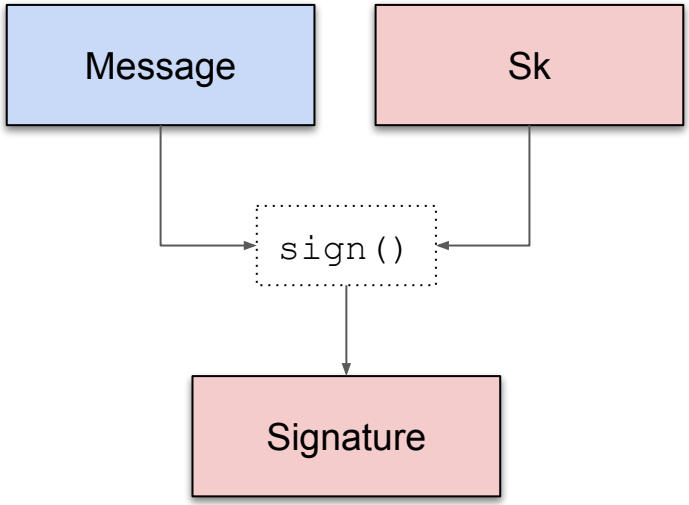
Problems:

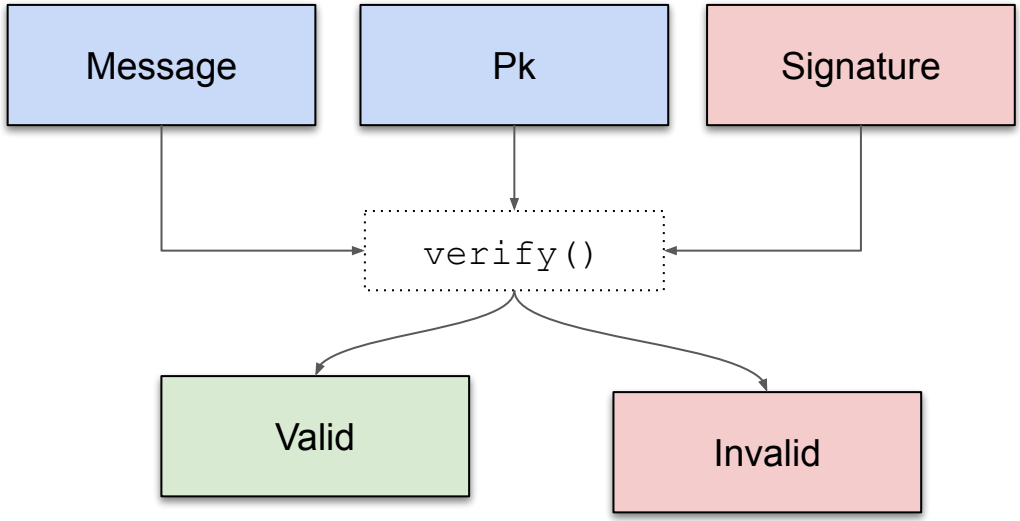
1. Authenticity of transactions
2. Privacy of transactions

✓ Co-location of participants and transaction transmission
✓ Learning the history of the ledger and trusting it
✓ It will be very slow because organizing and running marathons takes time
✓ What is the motivation to run marathons for others?

Cryptography in Blockchain

Public key = Account that identifies you (wallet address)
Private key = Cryptographic key that allows you to sign transactions





Digital signature in blockchain

Public key = Account that identifies you (wallet address)
Private key = Cryptographic key that allows you to sign transactions

Block 1

Prev: 0x001a3b
Tx1
Tx2...
Nonce: 2378001
Hash: 0x00fca5

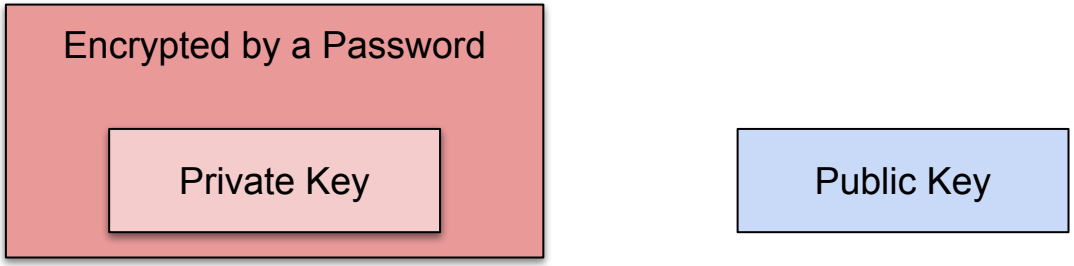
Transaction:

1. Source account (Pk)
2. Target account (Pk),
3. How much
4. Signature (using Sk of the source account)

Digital signature in blockchain

Private-public key pairs can be easily generated locally (RSA, ECDSA).

RSA: ssh-keygen
ECDSA for Ethereum: geth, MyEtherWallet

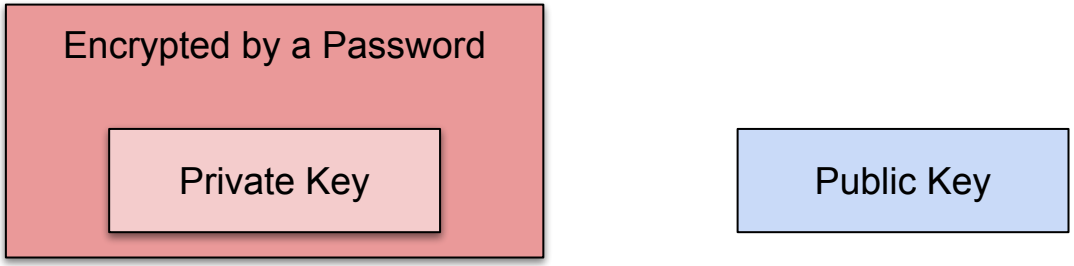


Everyone can obtain an account on blockchain!

Digital signature in blockchain

Private-public key pairs can be easily generated locally (RSA, ECDSA).

RSA: ssh-keygen
ECDSA for Ethereum: geth, MyEtherWallet



By replacing people's names with addresses, we introduced the concept of **pseudoanonymity**.

Proof-of-Work Blockchain

Choose leader by proof of work in a P2P network, use public keys as accounts, sign the transactions using private keys.

Problems:

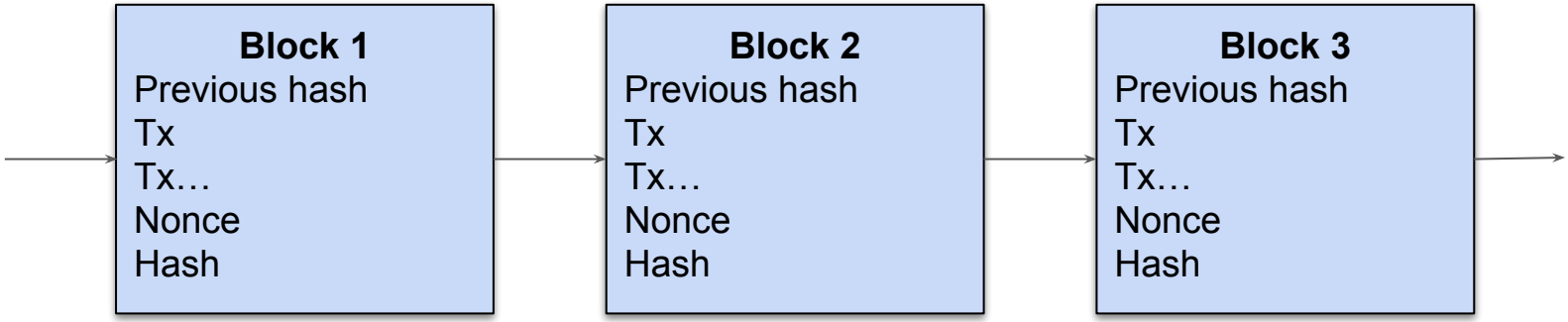
- ✓ Authenticity of transactions
- ✓ Privacy of transactions
- ✓ Colocation of participants in a single room
- ✓ Learning the history of the ledger and trusting it
- ✓ It will be very slow because organizing and running marathons takes time
- ✓ What is the motivation to run marathons for others?

From Blockchain to Computational Platform

Computing on blockchain

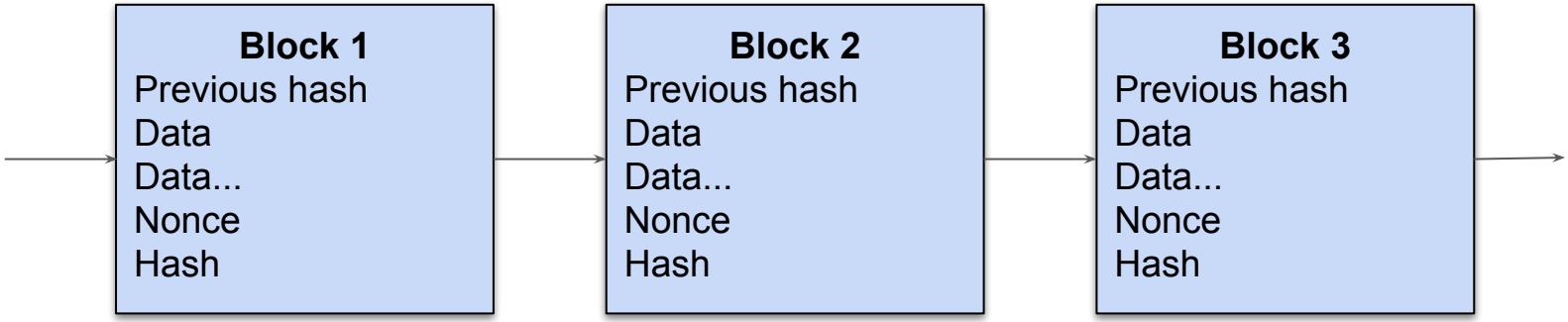
Transaction:

1. Source account (Pk)
2. Target account (Pk),
3. How much
4. Signature (using Sk of the source account)



Computing on blockchain

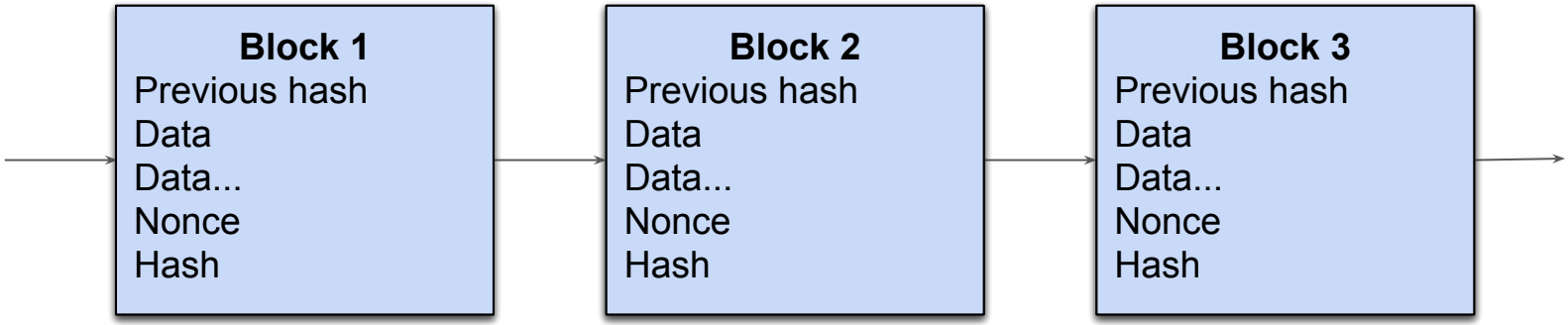
Key idea: Blockchain allows us to store data. This can be transactions or anything else!



Computing on blockchain

Key idea: Blockchain allows us to store data. This can be transactions or anything else!

Use the “Data” to store instructions of the program!



Computing on blockchain

Ethereum Virtual Machine (EVM): Stack-based virtual machine described by V. Buterin in 2013 and G. Wood in 2014 - 2019.

Smart contract: Program for the EVM.

1. Smart contracts expose API that can be called within transactions by Ethereum accounts

Computing on blockchain

Ethereum Virtual Machine (EVM): Stack-based virtual machine described by V. Buterin in 2013 and G. Wood in 2014 - 2019.

Smart contract: Program for the EVM.

1. Smart contracts expose API that can be called within transactions by Ethereum accounts
2. Smart contracts define what EVM opcodes should be executed by miners in order to update their chain state

Computing on blockchain

Ethereum Virtual Machine (EVM): Stack-based virtual machine described by V. Buterin in 2013 and G. Wood in 2014 - 2019.

Smart contract: Program for the EVM.

1. Smart contracts expose API that can be called within transactions by Ethereum accounts
2. Smart contracts define what EVM opcodes should be executed by miners in order to update their chain state

Executing each opcode costs **gas** which is used to reward the miners for using their computational power for the PoW and for maintaining the network state

Example of a smart contract

```
1.  contract BasicToken {
2.
3.    mapping(address => uint256) balances;
4.
5.    function transfer(address _to, uint256 _value)
6.    public returns (bool) {
7.        balance.balances[msg.sender] -= _value;
8.        balances[_to] += _value;
9.        return true;
10.   }
11.
12.  ...
```

Examples of Vulnerabilities

Live Demo

Re-entrancy: A method is called **re-entrant** if it can be interrupted in the middle of its execution and then safely called again ("re-entered") before its previous invocations complete their executions.

Live demo in Remix

Vulnerability Categories in Ethereum

Solidity	EVM	Blockchain
Reentrancy	Immutable bugs	Unpredictable state
Call to unknown	Ether lost in transfer	Generating randomness
Gasless send	Stack size limit	Time constraints
Exception disorders	Integer overflows	**Transaction ordering**
Type casts		Network throughput
		Keeping secrets

FairWin


Equitable dividend-sharing game with payment in ethernic currency
Provable fairness The dividend-sharing game is simple Open Source Contracts Provide Support
All Wagers
703610ETH
58914bets

FairWin

1. A ponzi scheme deployed on July 27, 2019.
2. On September 26, it held ~49,518 ETH.
3. Vulnerabilities were found and reported in September 2019.
4. The contract is currently empty.

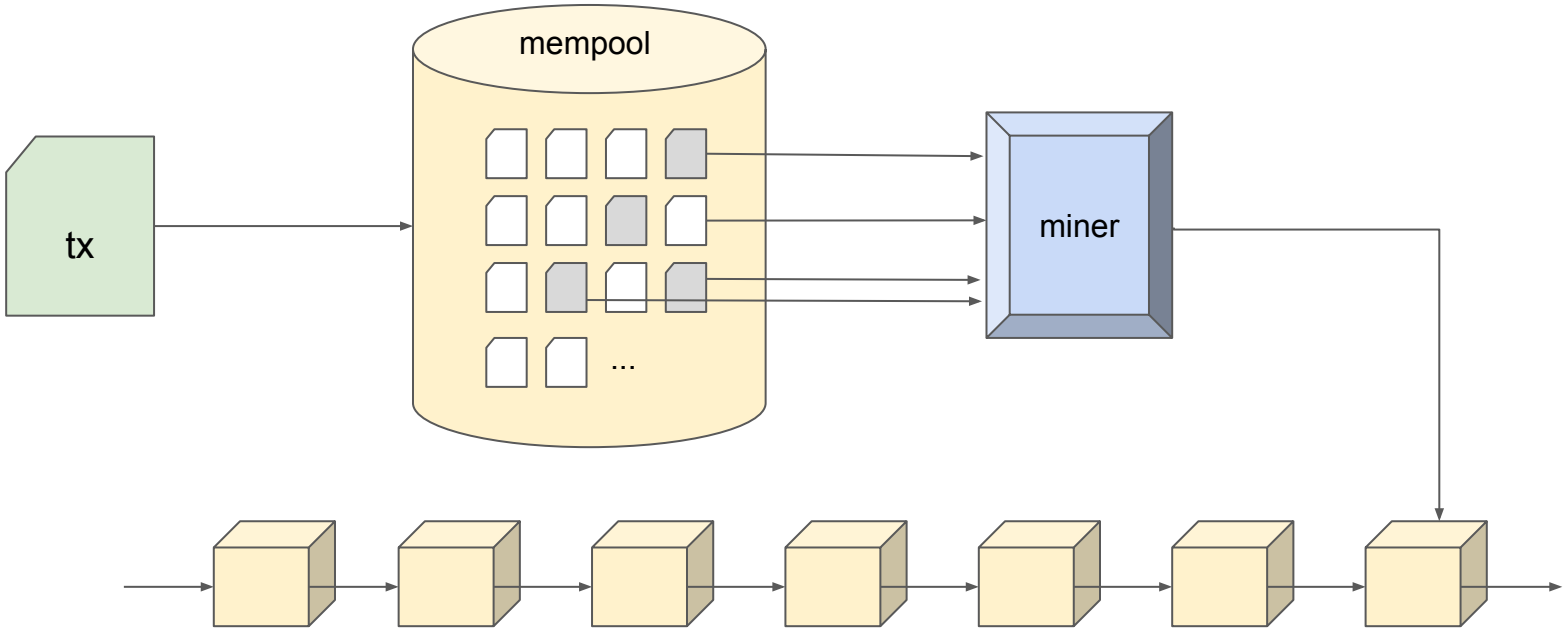
FairWin


FAIRWIN.ME
Marshall
initiators of the FAIRWIN
FAIRWIN Public Winning Intelligent Contract Game
FAIRWIN公赢智能合约游戏

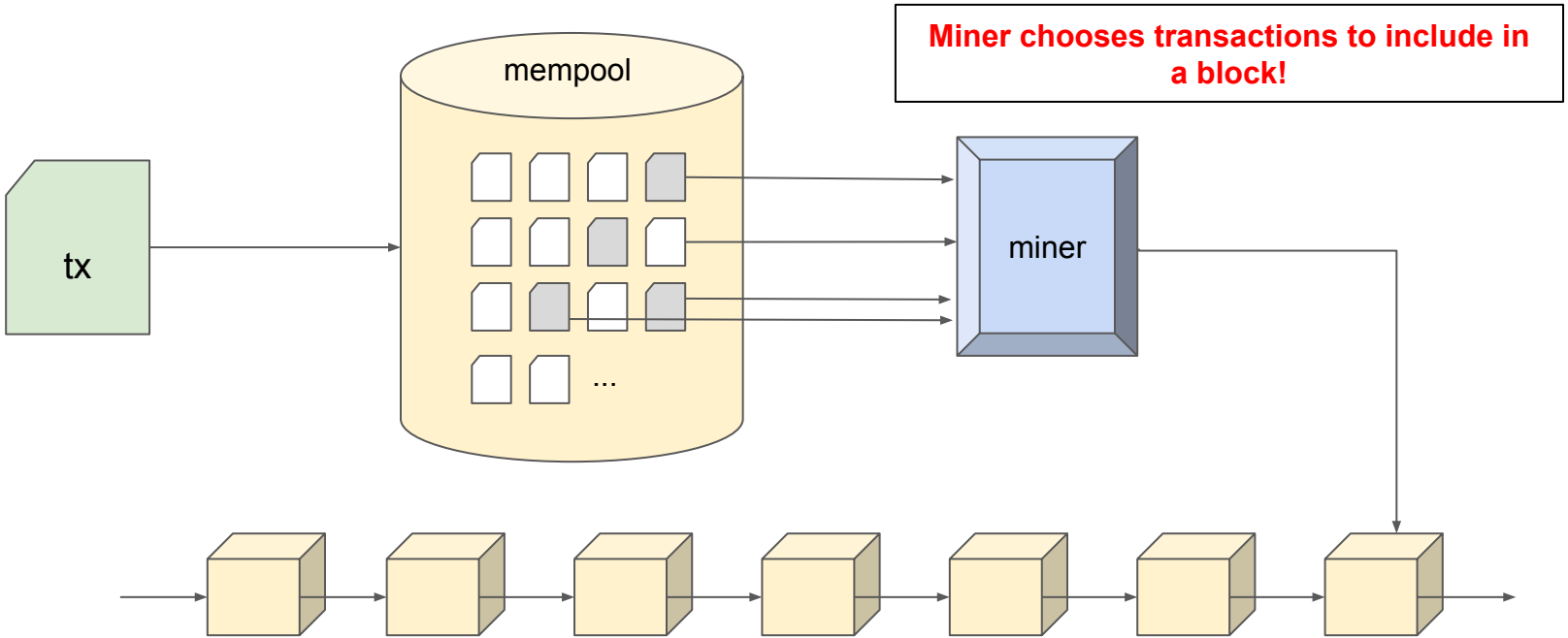
Simplified Code

```
contract FairWin {

  mapping (string => address) investors;
  mapping (string => uint256) investments;

  function invest(string inviteCode, ...) payable {
    if (investors[inviteCode] == 0x0) investors[inviteCode] = address;
    investments[inviteCode] = investments[inviteCode].add(msg.value);

    //… other logic
  }

  function exit(string inviteCode) {
    require(investors[inviteCode] == msg.sender);
    investors[inviteCode].send(investments[inviteCode]);
  }
}
```

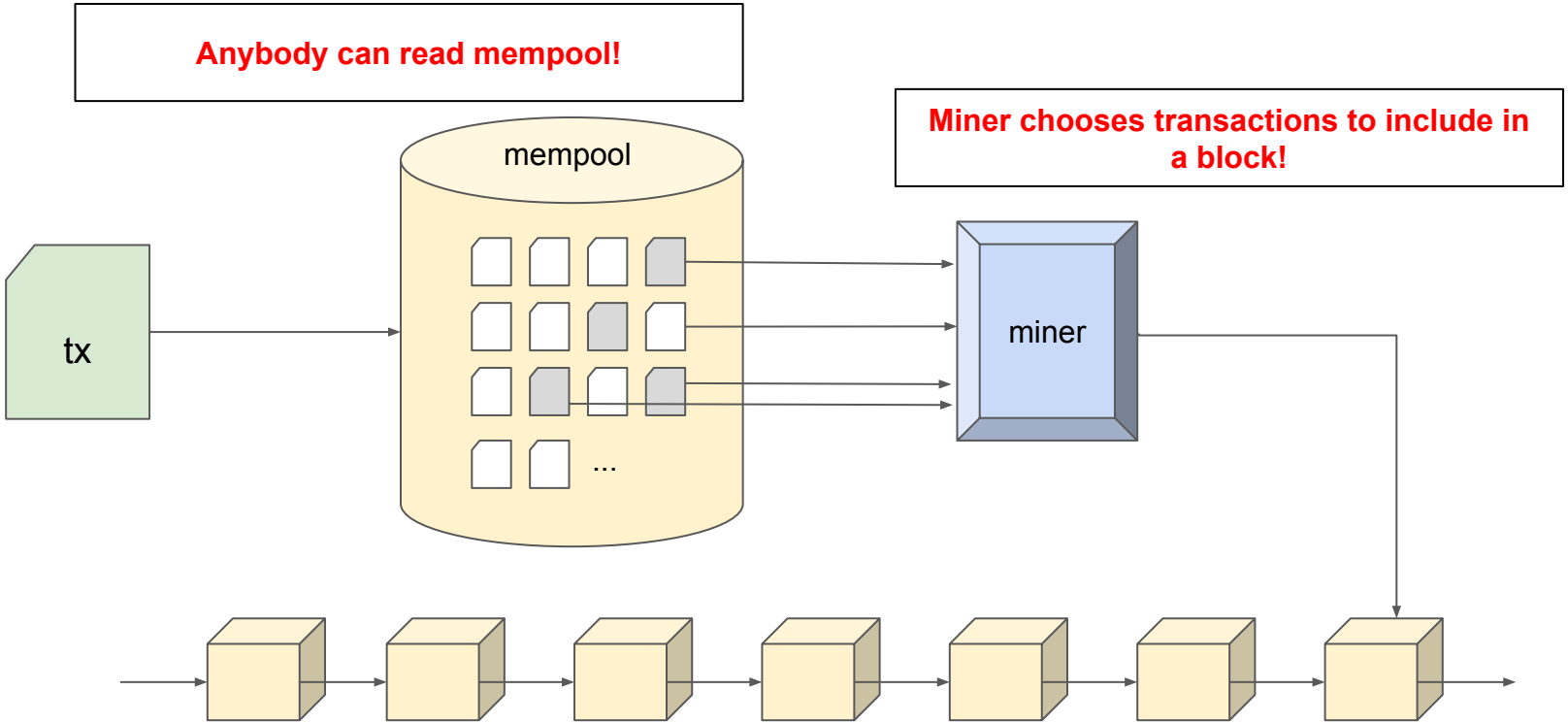
Back to Proof-of-Work



Back to Proof-of-Work



Back to Proof-of-Work



Research Topics

Research Areas

Scaling blockchain

- Proof-of-Work takes time to produce (it cannot be too easy!)
- The number of transactions per minute can be low
- Proof-of-Work is far from being green

Possible solutions

- Different type of consensus (Proof-of-Stake, Proof-of-Authority, Algorand)
- Side channels (plasma, chain sharding)
- Combination of on-chain and off-chain approaches

Research Areas

Smart contract and platform implementation

- Every opcode execution costs gas
  - *Perez, Livshits (2019): Broken Metre: Attacking Resource Metering in EVM*
- Traditional algorithms are often insufficient
  - Long iterations are costly
  - Updates to storage is expensive
  - Array lists are not usable
  - There is no primitive for generating (pseudo)random numbers
- Blockchain is not a database
  - Unpredictable availability, unpredictable rollbacks

Research Areas

Tools for smart contract analysis

- Static analysis and symbolic execution (Mythril, Oyente, Slither, Securify)
- Formal verification tools (K-Framework)
- Compiler verification tools
- EVM verification tools
- New programming languages

Research Areas

Tools for smart contract analysis

- Static analysis and symbolic execution (Mythril, Oyente, Slither, Securify)
- Formal verification tools (K-Framework)
- Compiler verification tools
- EVM verification tools
- New programming languages

Verification of off-chain computations

- SNARKs
- STARKs
- BulletProofs

Conclusions

- **Blockchain is NOT a new technology**
  - Orchestration of known techniques in a new way
  - Using it is prone to making mistake

- **Social revolution**
  - Redistribution of wealth
  - Means of spreading access to services
  - New field for tech business
  - DeFi (decentralized finance)

Conclusions

- **Popularization of computer science**
  - LaTeX is cool
  - Turing completeness is cool
  - Randomness, probability and cryptography is cool
  - Algebra and game theory is cool

Conclusions

- **Popularization of computer science**
  - LaTeX is cool
  - Turing completeness is cool
  - Randomness, probability and cryptography is cool
  - Algebra and game theory is cool

Thank you!