Blockchain

Nick Kozlov - MSP (Rus)
Nikita.Kozlov@studentpartner.com

Dasha Slesareva
sls-daria99@yandex.ru

Blockchain

Why is it so popular?

What will it change?

Technology aspects

You will understand how it works

Practice

We will make our personal smart contract and more

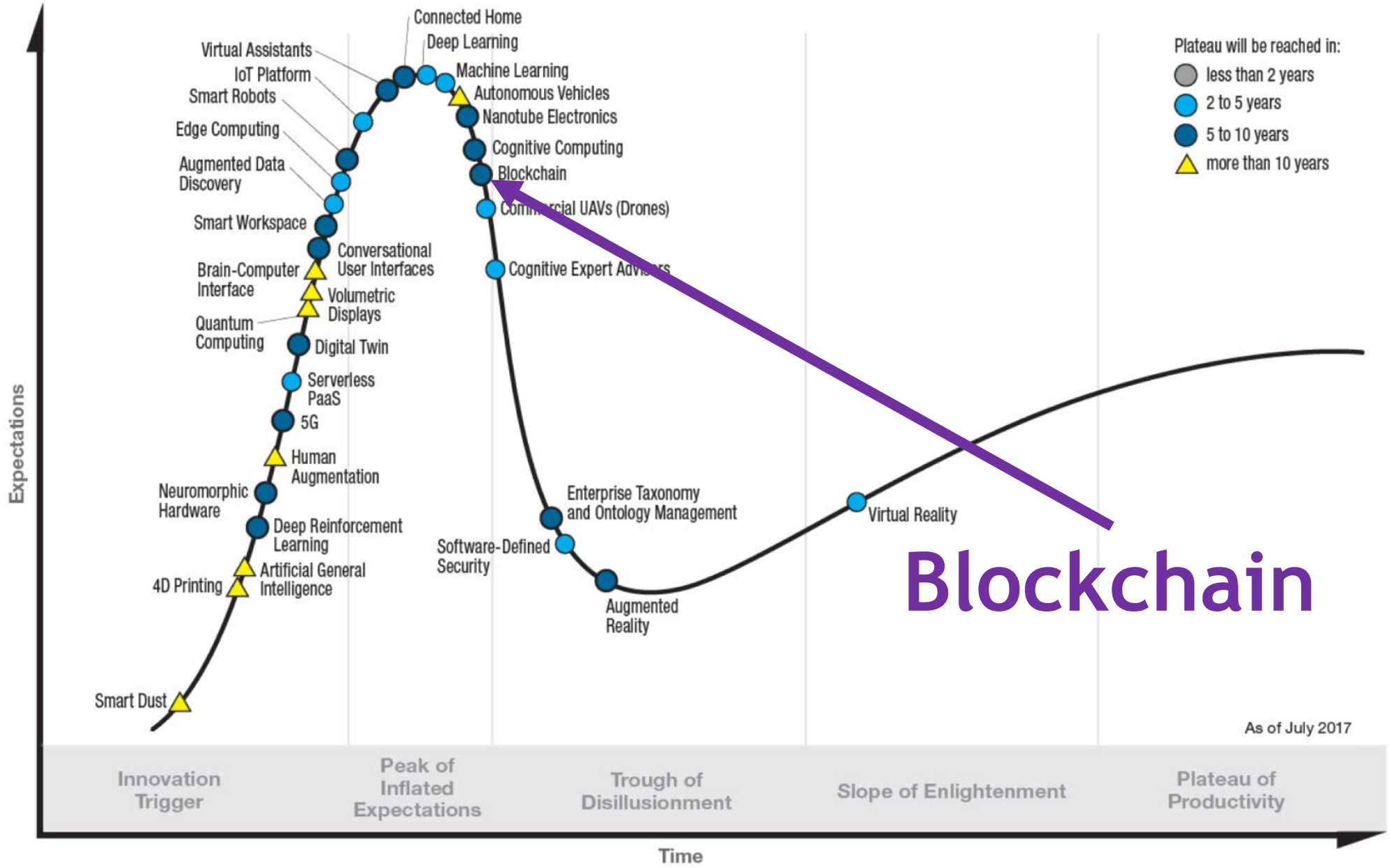
Blockchain

It will save more than 40 billion $

The next wave of technology revolution

Reduce the difference between rich and poor people

Gartner Hype Cycle for Emerging Technologies, 2017



Mainframe

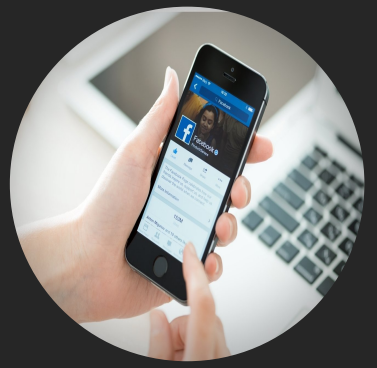
1980s

eCommerce

2000s

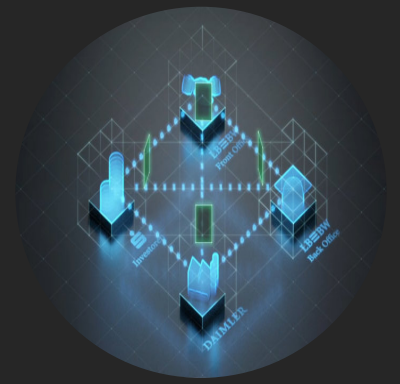
PC

1990s



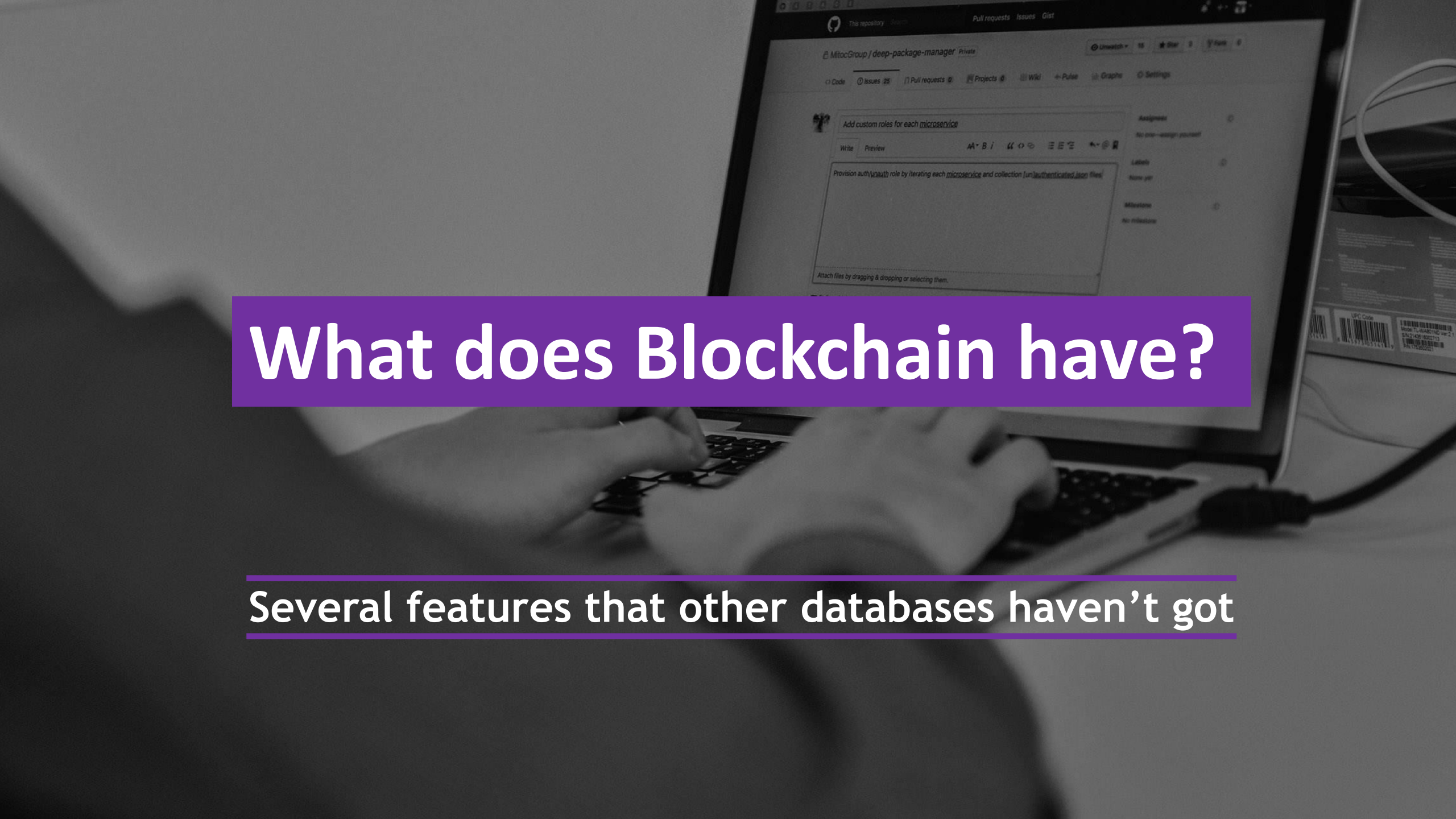
Mobile / Social

2010s



Internet of value

2020s



What does Blockchain have?

Several features that other databases haven’t got

Each immutable entry is written once, thereby increasing visibility and auditability, while reducing error rates

Authenticated counterparties digitally sign requests, updates and claims

Secure

Shared

Distributed

Authoritative

Each member of the network can use the Blockchain to validate the other counterparties

Applicants and beneficiaries collaborate in near real-time using standardized templates

Blockchain is a shared secured and distributed registry

Confirmed by encryption

Is based on reliable technology of signatures with public and private keys.

It allows you to create transactions in Blockchain that are protected what helps to establish a common trust

Blockchain is a shared secured and distributed registry

General access

The value of Blockchain technology is directly proportional to the number of organizations and companies using it.

Even in conditions of severe competition it is profitable for competitors to participate together in the deployment of this shared distributed database

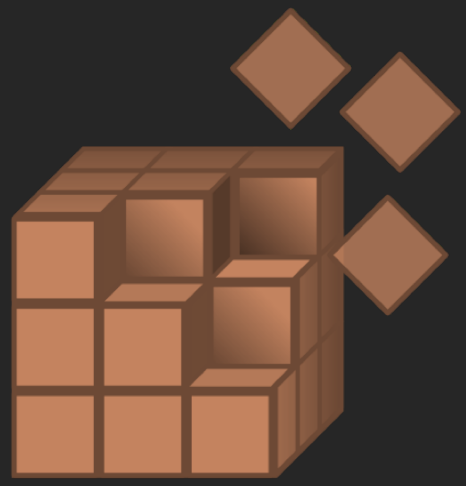
Blockchain is a shared secured and distributed registry

Distributed architecture

There are many lines of the Blockchain database. In fact, the are more lines, the higher the reliability of the data is

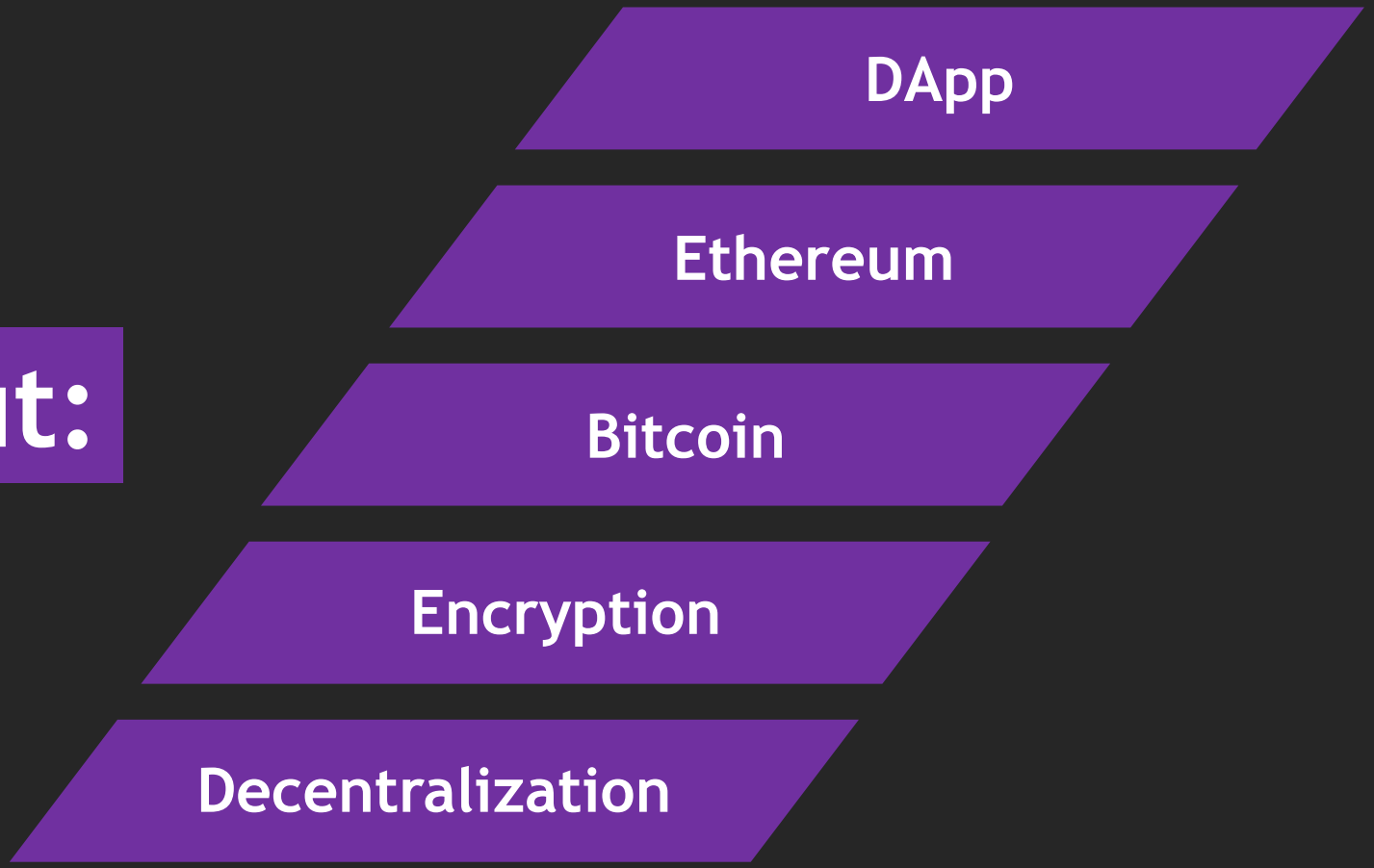
Blockchain is a shared secured and distributed registry

The Registry



The database has read and write access => it permanently records all transactions

We will talk about:



Decentralization

Decentralization

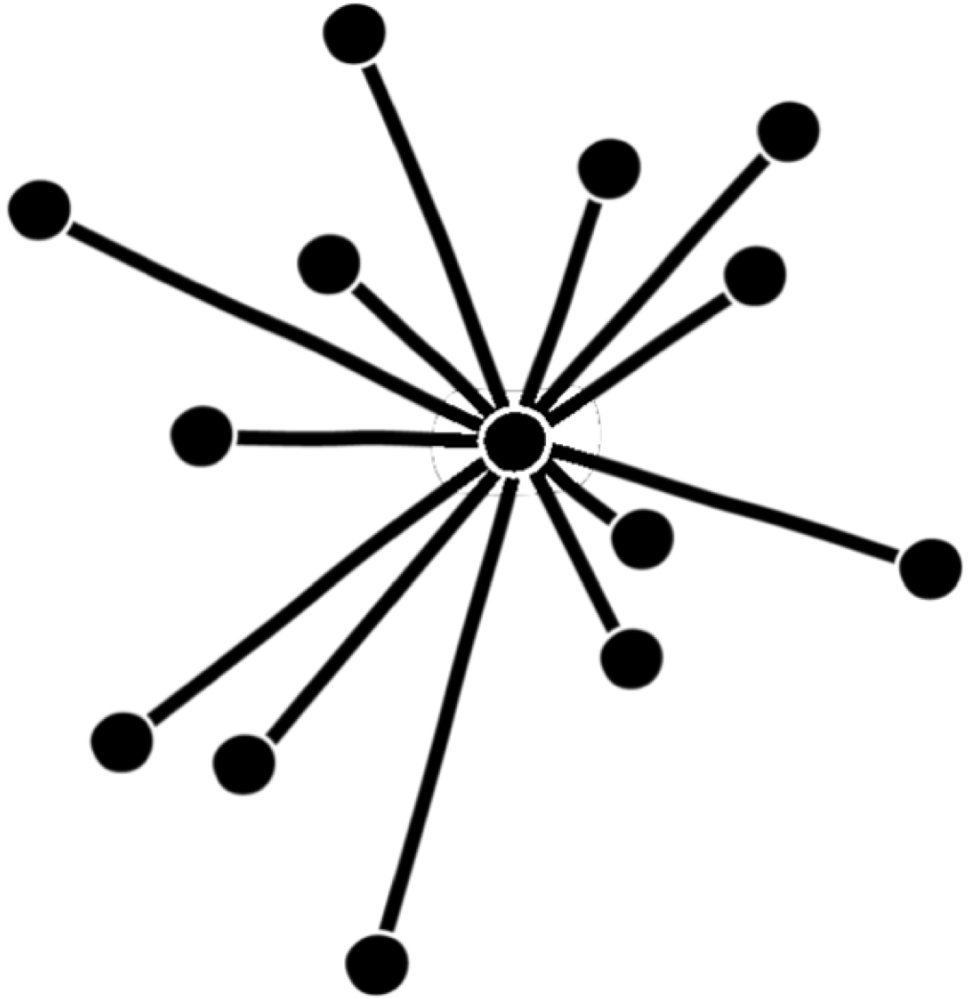
Centralized system

Decentralized system

Decentralization

Centralized system

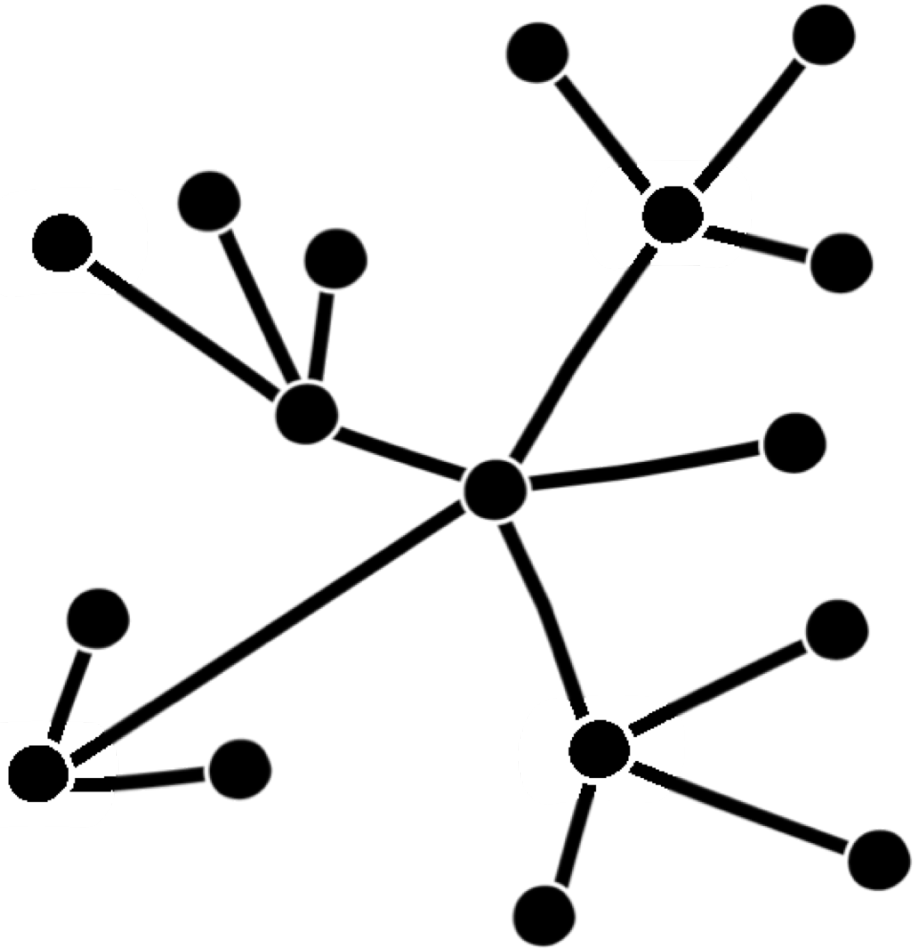
- ❑ One control point
- ❑ One point of failure
- ❑ One point of trust
- ❑ One point of attack
- ❑ Single bottleneck system



Decentralization

- ❑ We don‘t have control center
- ❑ Based on agreement of participants

Decentralized system



Decentralization is good

Decentralization is good

However…

If we don‘t have Central Authorities

We need to create a trust protocol

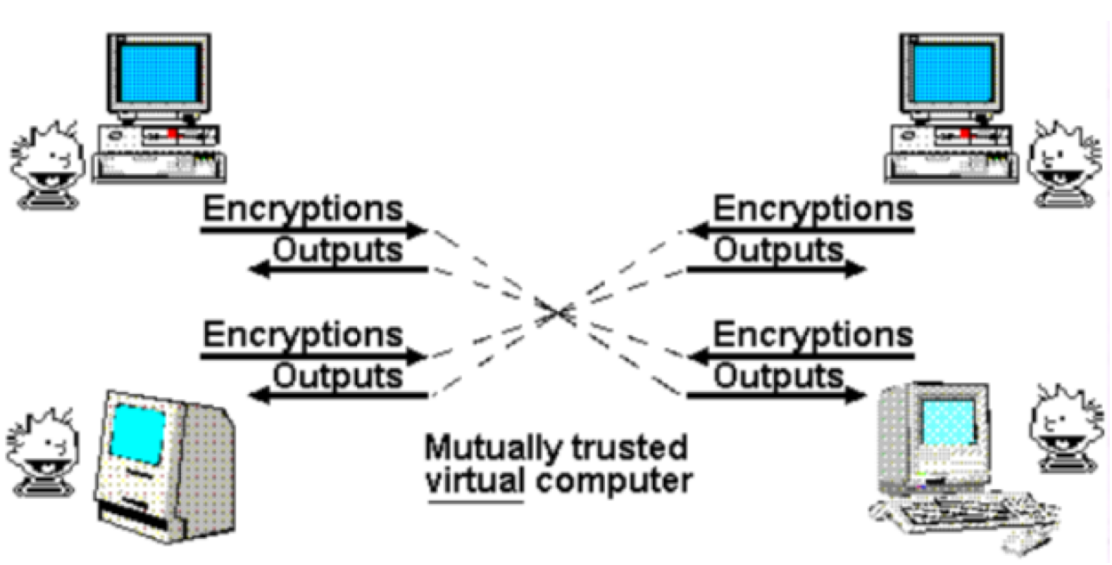
Without trusted 3rd parties

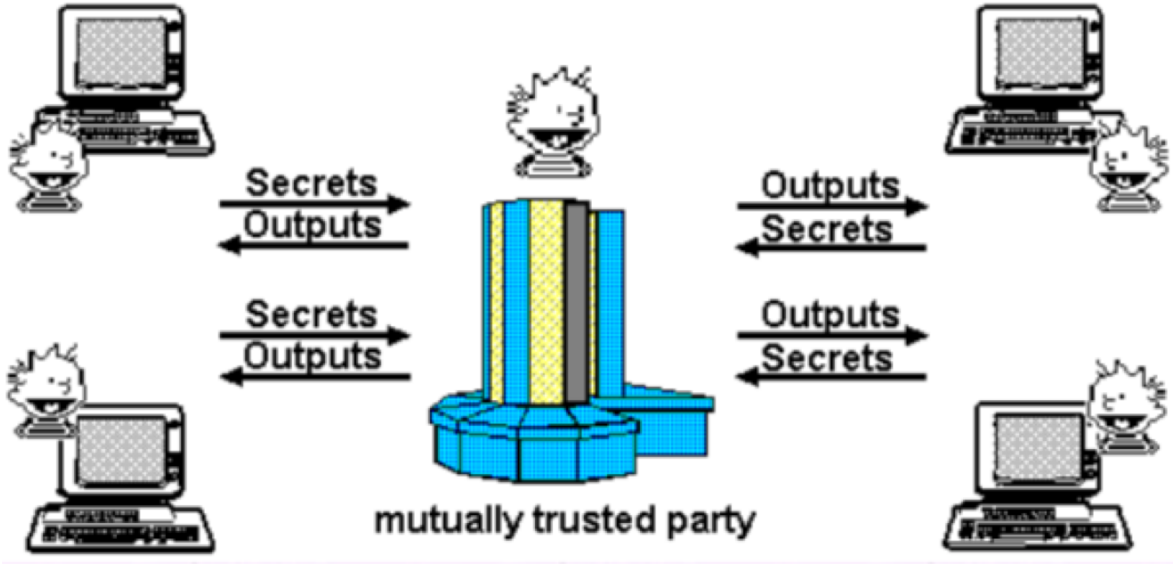
Who will keep the transaction log?

Who decides which transactions are valid?

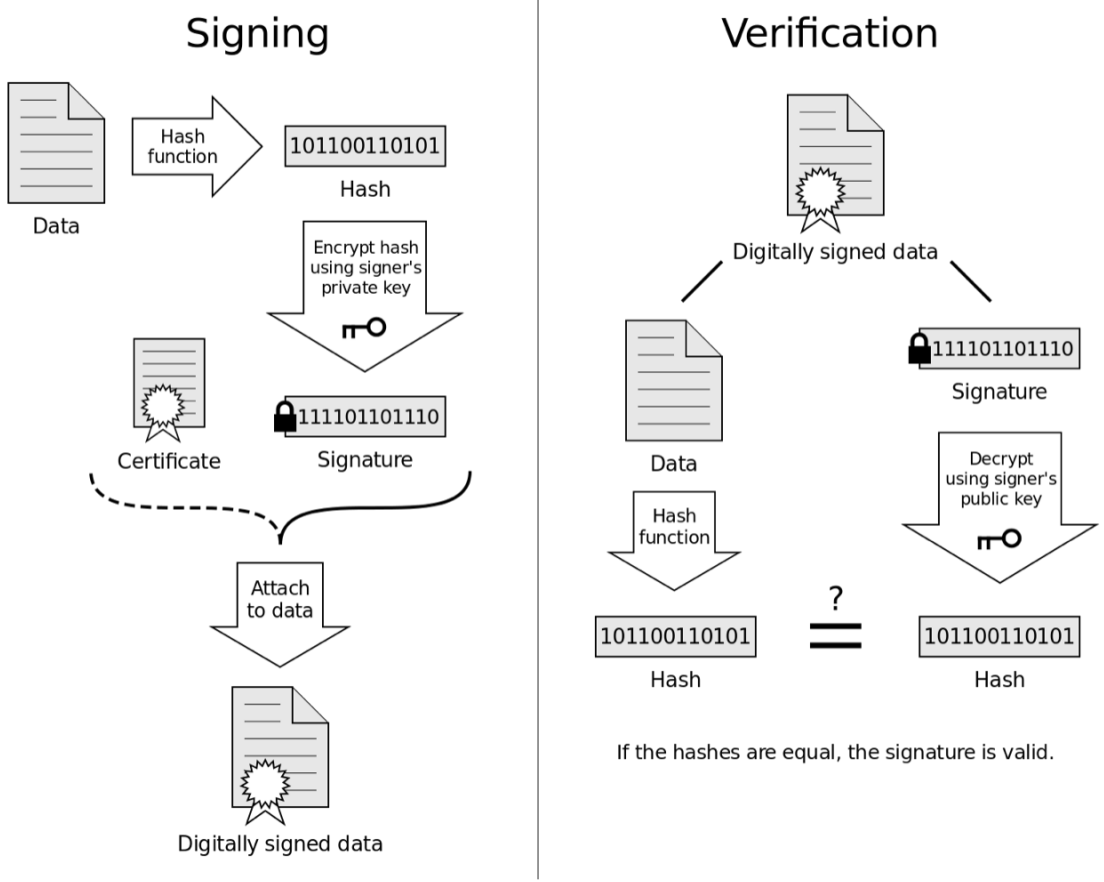
How to reach agreement in decentralized network?

Nick Szabo – “The God Protocol”





Solution : we replace trust by encryption



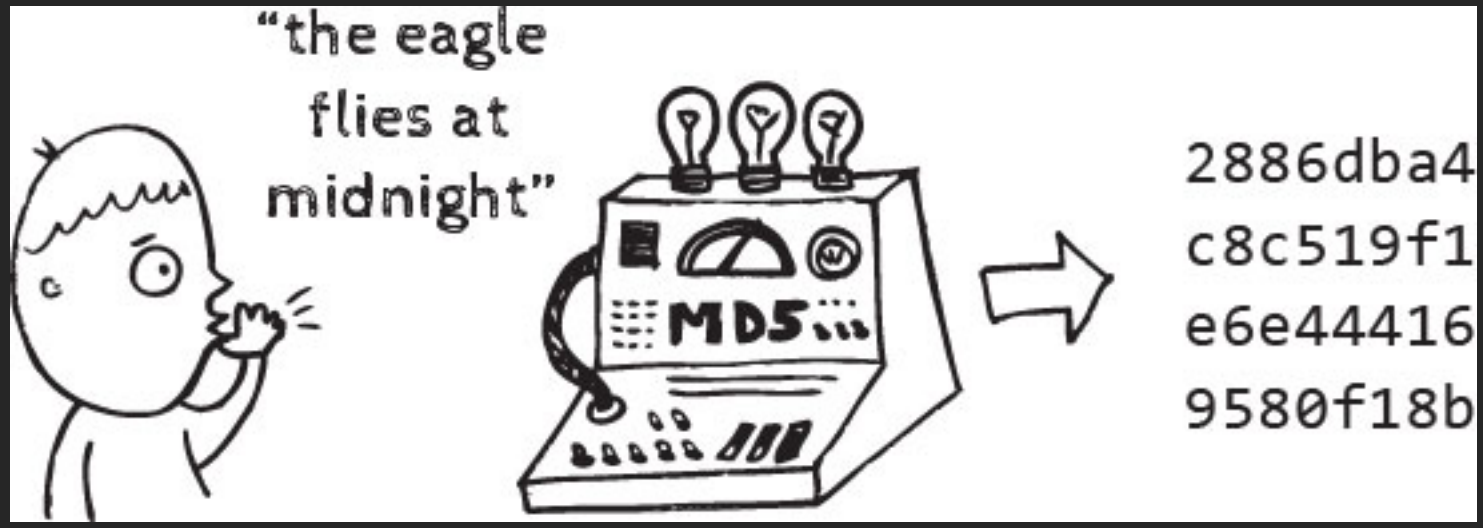
We need a payment system which will be based on encryption instead of trust

It allows anybody to make transaction without 3rd parties.

Questions

Encryption

Hash functions



Hash functions

- Deterministic transformation of any size data (input data) into fixed-size data (hash sums)

Hash functions

- Deterministic transformation of any size data (input data) into fixed-size data (hash sums)
- Hash *h* it should be hard to find any message m such that h=hash(m)

Hash functions

- Deterministic transformation of any size data (input data) into fixed-size data (hash sums)
- Hash *h* it should be hard to find any message m such that h=hash(m)
- Input m1, it should be hard to find another input, m2 != m1 such that hash(m1)=hash(m2)

Hash functions

- Deterministic transformation of any size data (input data) into fixed-size data (hash sums)
- Hash *h* it should be hard to find any message m such that h=hash(m)
- Input m1, it should be hard to find another input, m2 != m1 such that hash(m1)=hash(m2)
- No information about input data

Encryption

*SHA256

Hash functions

Hello → 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

hello → 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824

2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824 → d7914fe546b684688bb95f4f888a92dfc680603a75f23eb823658031fff766d9

Hello HSE students → ea526da0ea9e2f472afbdb647ffad0cd63567f59fb9a4bfa12429f5dbfed5c1e

Encryption

*SHA256

Hash functions

Hello 

hello 

16f6759caac9e976cfbd38fefc5ec35e7e114e7512fbe9940dd5093783c38edc

2cf24dba5fb0a30e26e83b2ac5b9
e29e1b161e5c1fa7425e7304336
2938b9824 

Encryption

*SHA256

Info (i) → Hash function (H) → Hash code (i_{hash})

$$H(i) = i_{hash}$$

Encryption

*SHA256

Why Hash functions?

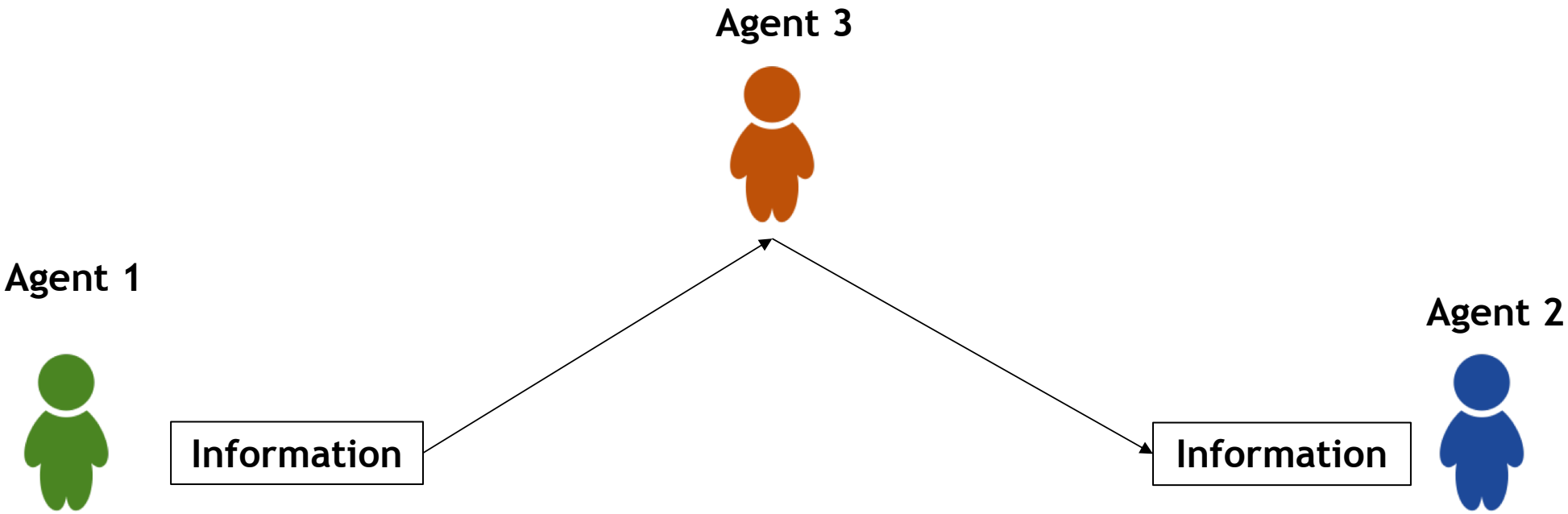
It helps to validate data

Content identification

Identification case

Public-key encryption

Case: we need to safely send data to agent 2 when we have agent 3?



Public-key encryption

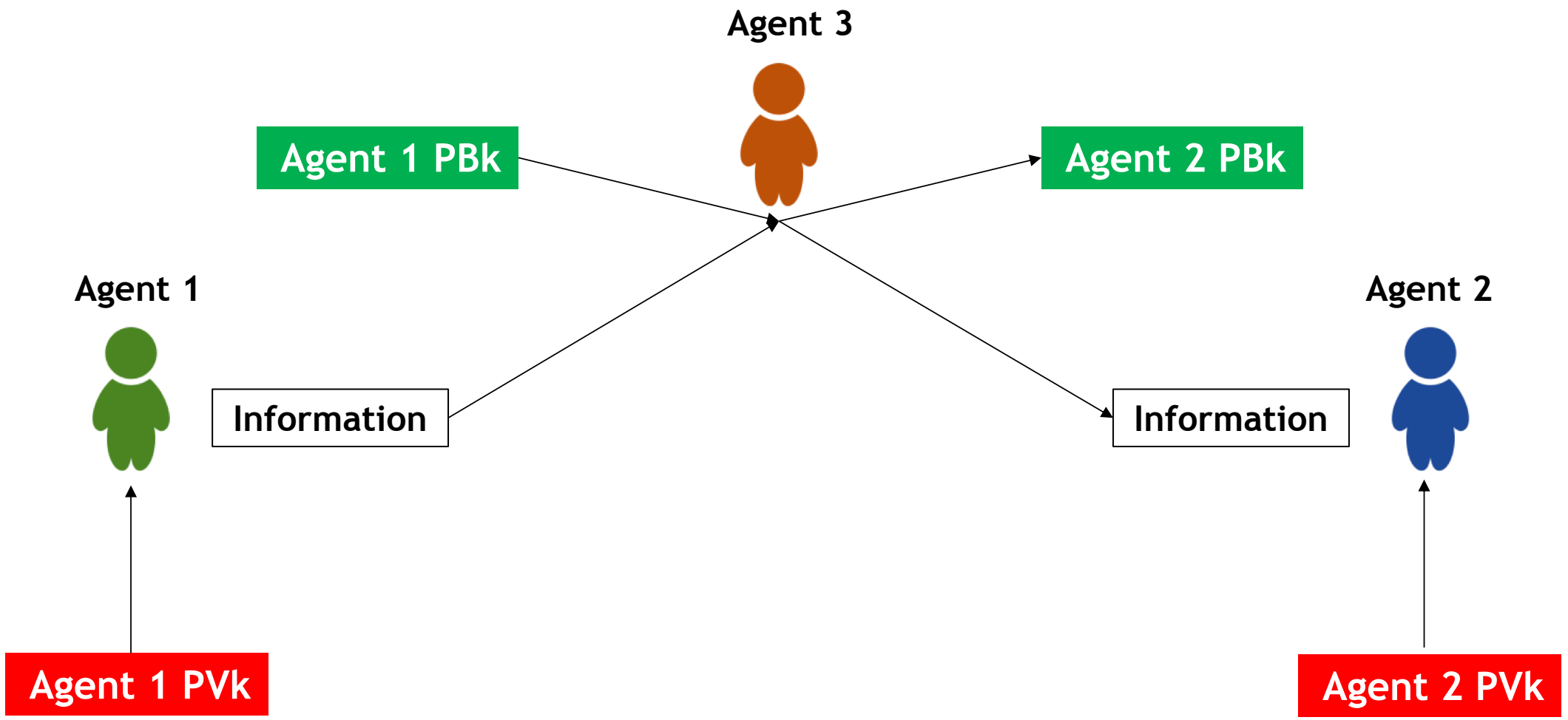
We will use keypairs

Private key is based on Public key

No data about Private key

Public-key encryption

Man in the middle problem



Public-key encryption

Agent1 Side:

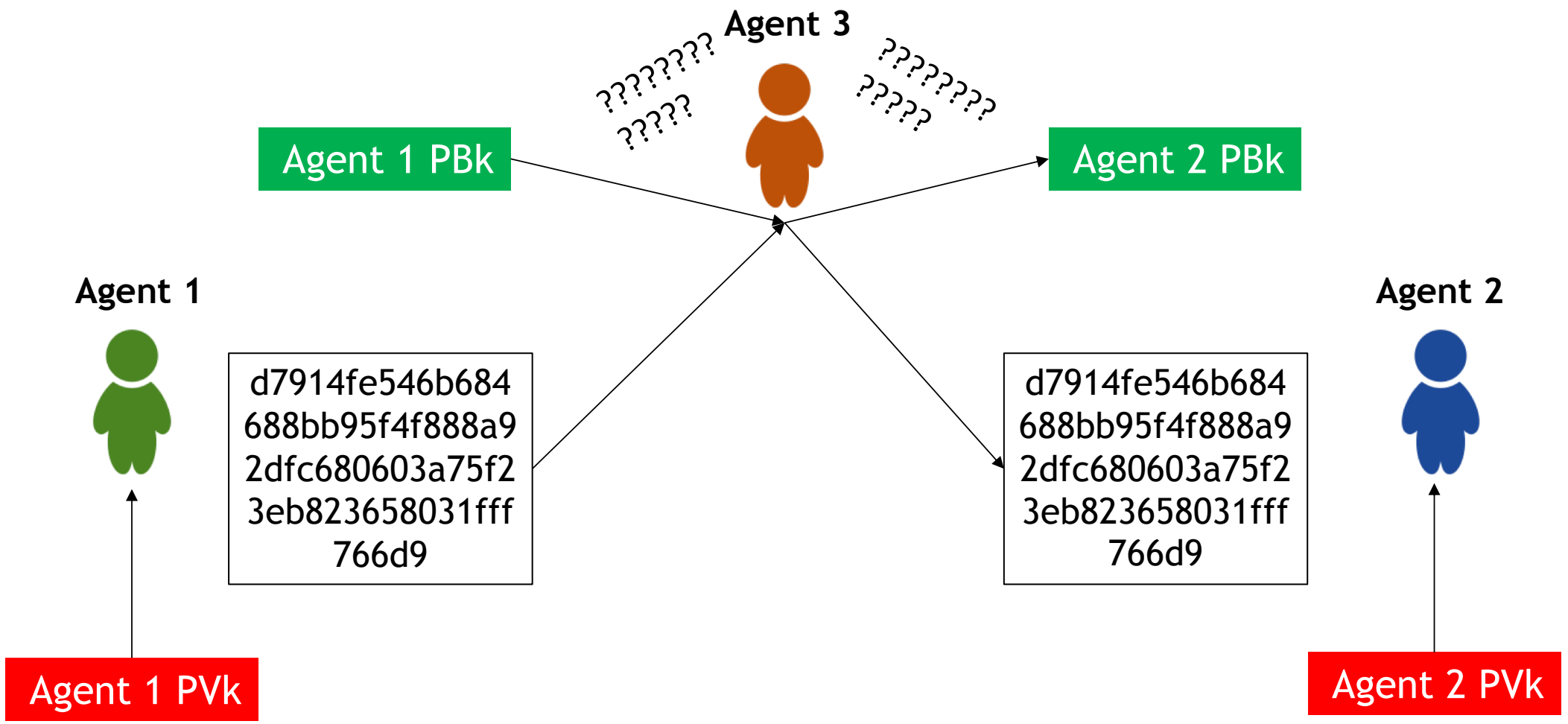
Information

Agent 1 PBk

d7914fe546b684
688bb95f4f888a
92dfc680603a75f
23eb823658031f
ff766d9

Public-key encryption

Solved!



Public-key encryption

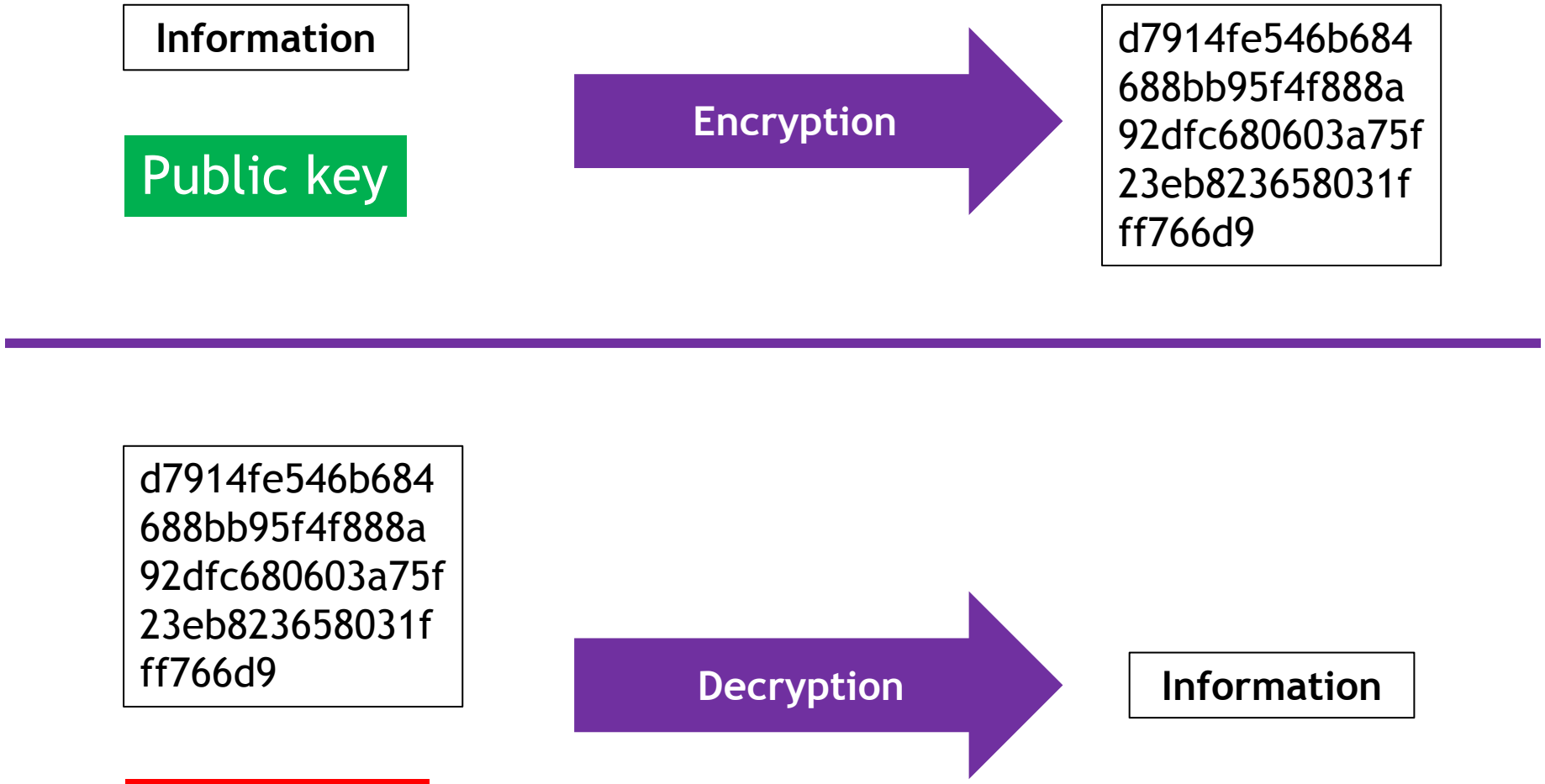
Agent2 Side:

d7914fe546b684
688bb95f4f888a
92dfc680603a75f
23eb823658031f
ff766d9

Agent 2 PVk

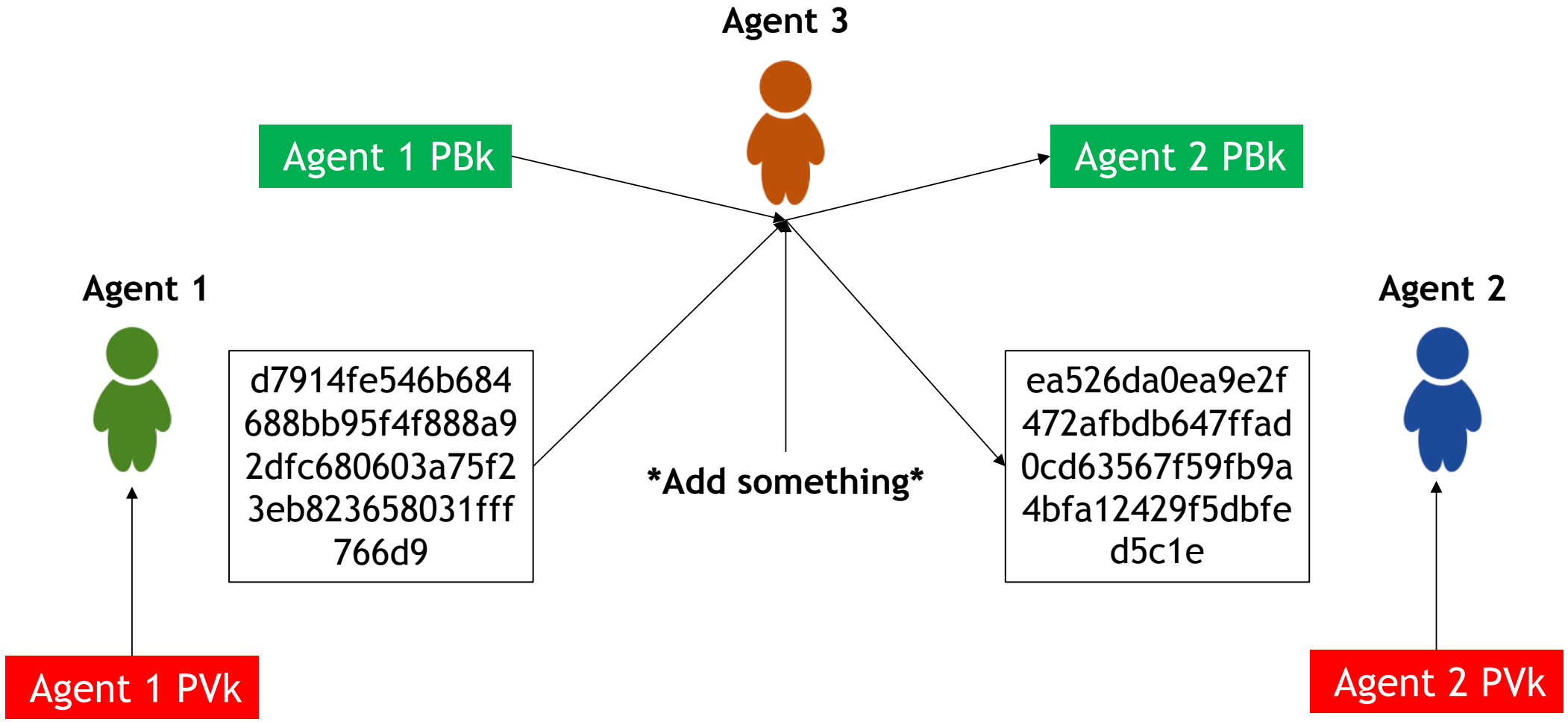
Information

Generally

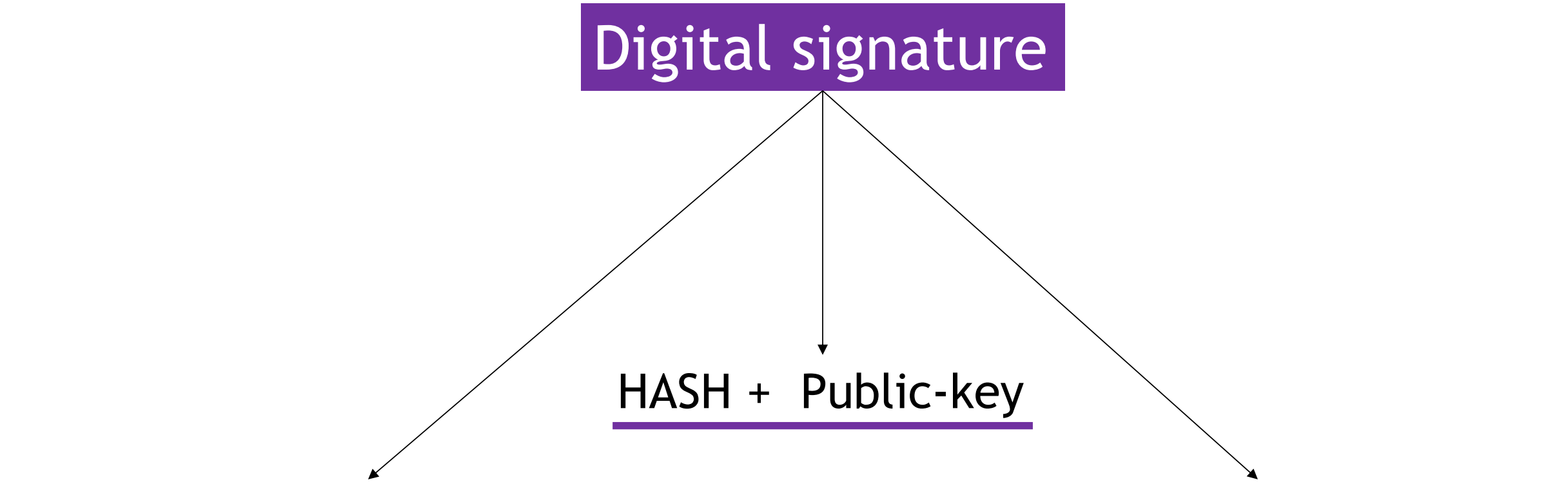

Information
Public key
Encryption
d7914fe546b684
688bb95f4f888a
92dfc680603a75f
23eb823658031f
ff766d9
d7914fe546b684
688bb95f4f888a
92dfc680603a75f
23eb823658031f
ff766d9
Decryption
Information
Private key

Public-key encryption with digital signature

Case: we need to safely send data to agent 2 when we have agent 3? AND be sure that it‘s correct

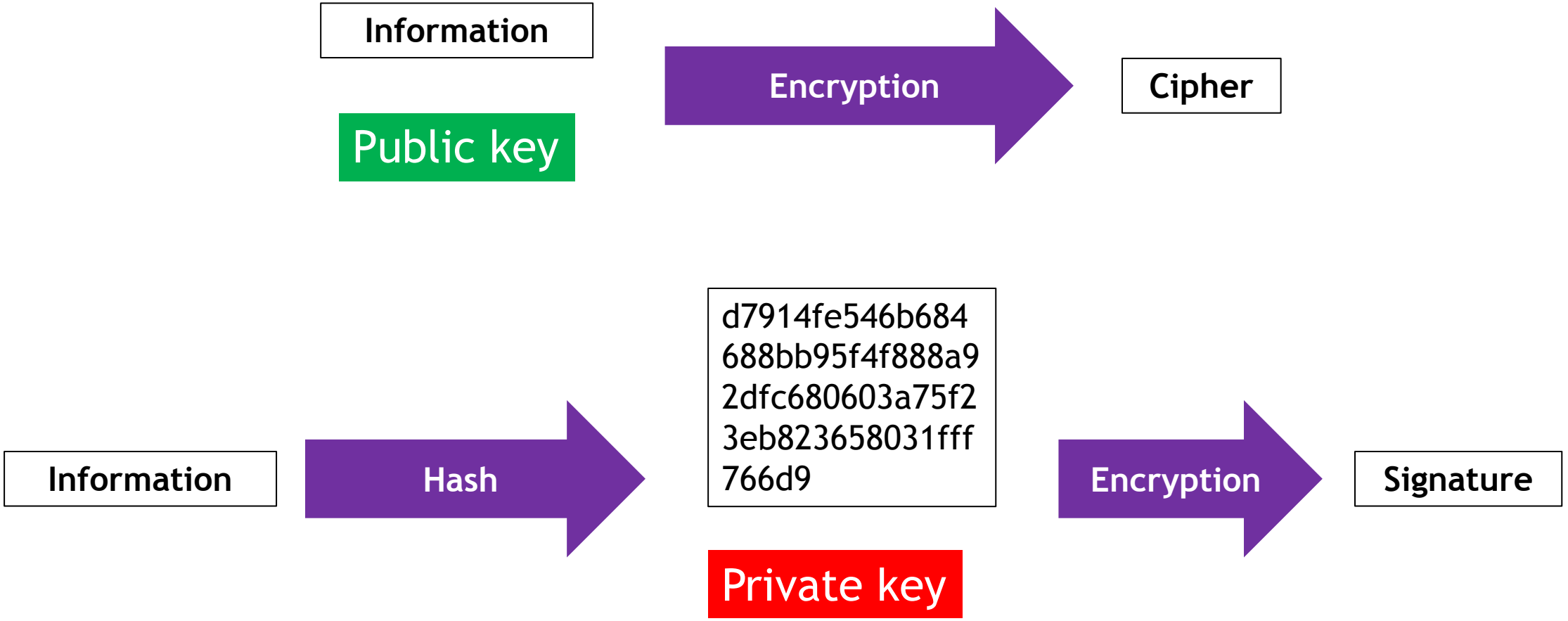


Public-key encryption with digital signature



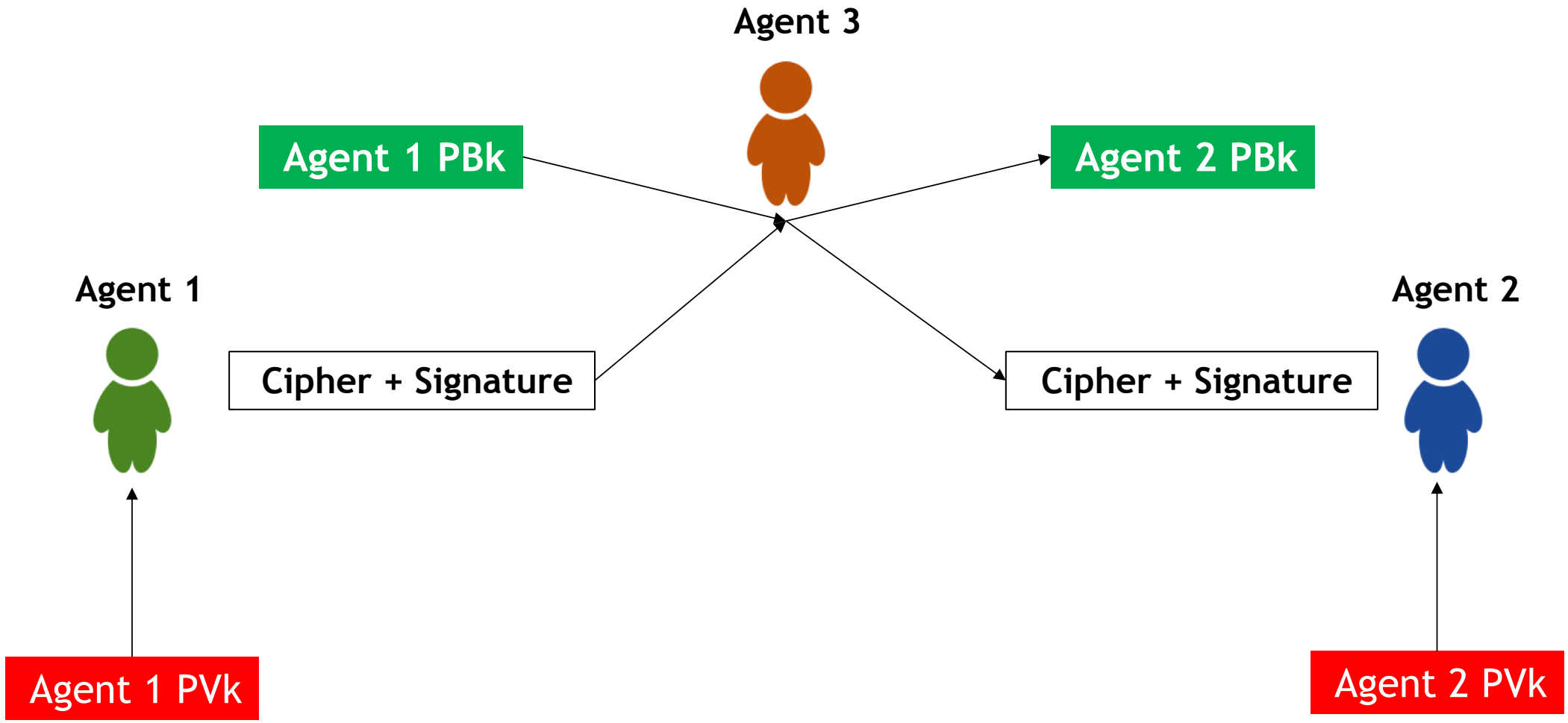
Public-key encryption with digital signature

Agent 1: making signature



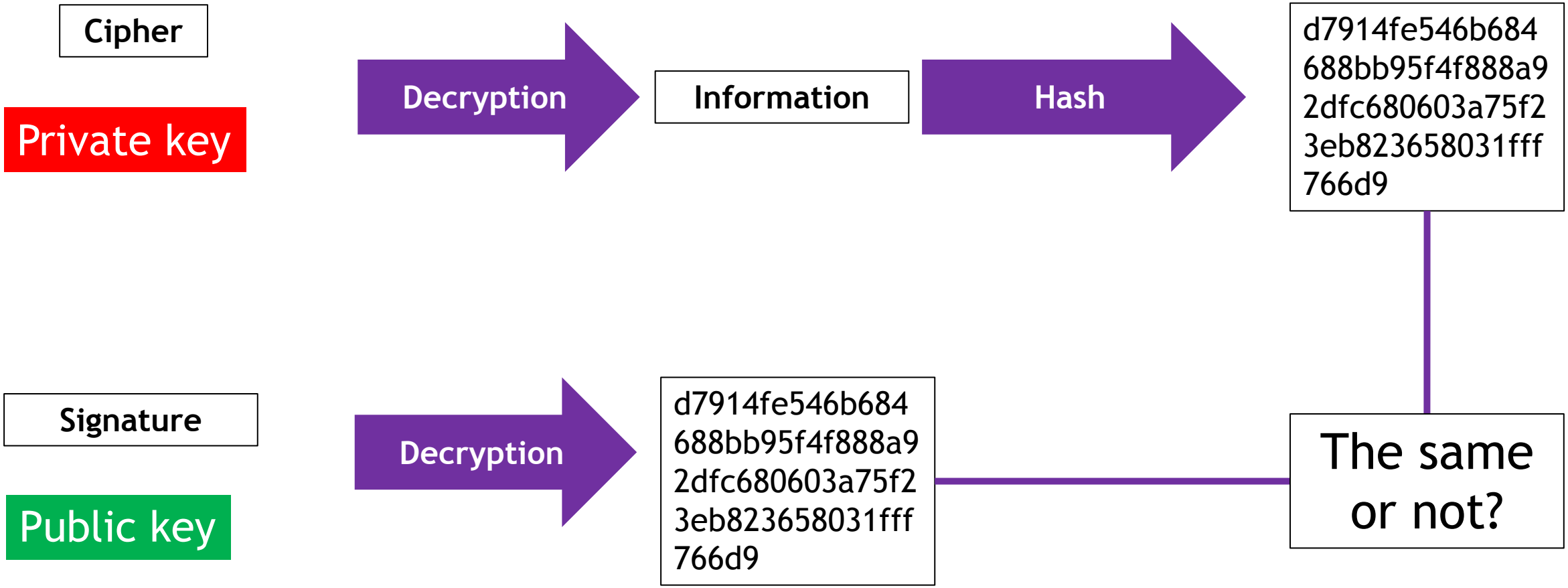
Public-key encryption with digital signature

Communication with digital signature



Public-key encryption with digital signature

Agent 2: checking process



Public-key encryption with digital signature

Attack scenarios:

1) Agent 3 will change information

2) Agent 3 will change information with his private key

1)

Signature != Cipher

2)

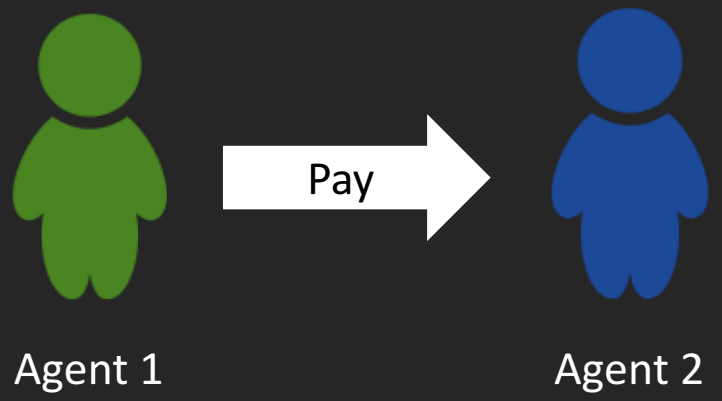
Decryption is impossible
(private key is not the same)

Questions

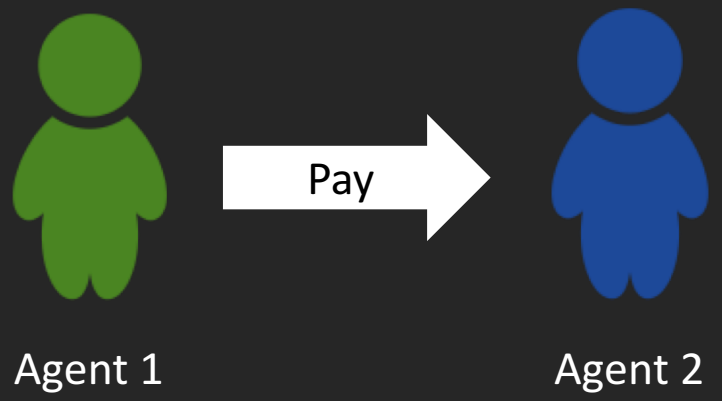
Bitcoin

Bitcoin

Let‘s make HSEcoin



- **We use HSEcoins**



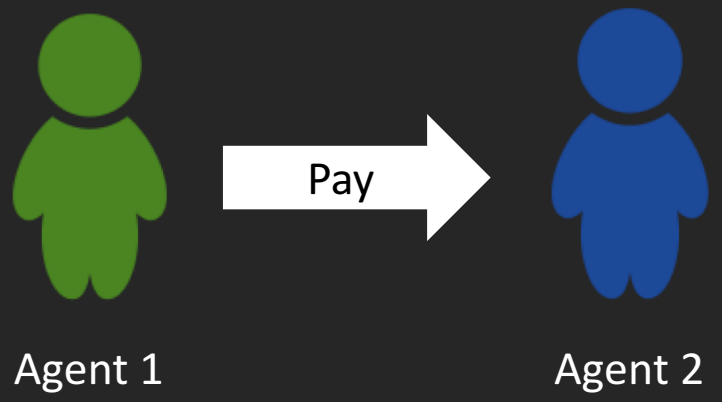
How can we prevent Agent 1 from using the same bit string over and over, thus minting an infinite supply of money?

How can we prevent someone else forging such a string of bits, and using that to steal from Agent 1?

Agent 1 PVk

+

Agent 1 send Agent 2
1 HSEcoin



We can be sure that Agent1 really wants to send Agent2 a HSEcoin

Agent 1 PVk

+

Agent 1 send Agent 2
1 HSEcoin

The bank updates their records to show that the HSEcoin with that serial number is now in Agents 2 possession, and no longer belongs to Agent 1

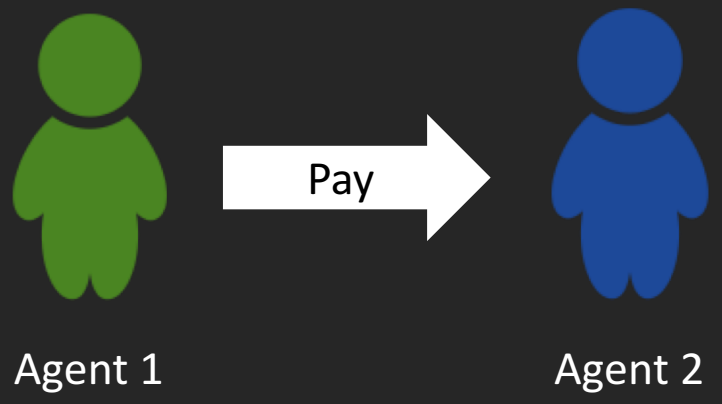
Agent 1 PVk

+

Agent 1 send Agent 2
1 HSEcoin

Making everyone collectively the bank

Transactions log
+
Serial number 1234567
+
Agent 1 PVk
+
Agent 1 send Agent 2 1 HSEcoin



Transactions log

We'll assume that everyone using HSEcoin keeps a complete record of which HSEcoins belong to which person. You can think of this as a shared public ledger showing all HSEcoin transactions. We'll call this ledger the block chain, since that's what the complete record will be called in Bitcoin, once we get to it.

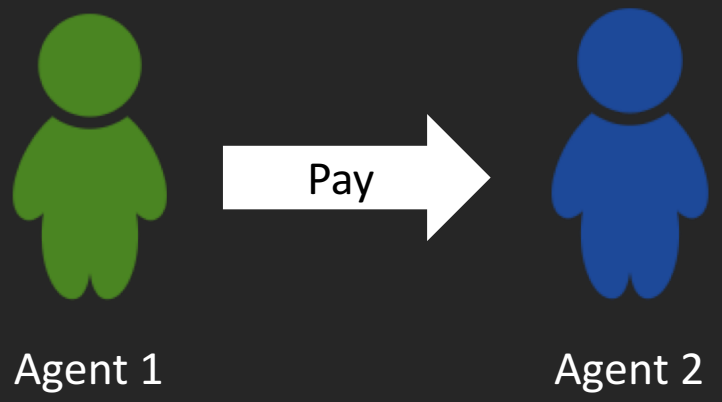
Serial number 1234567

+

Agent 1 PVk

+

Agent 1 send Agent 2 1 HSEcoin



Transactions log

How we make a serial number?
“where do serial number come from” problem

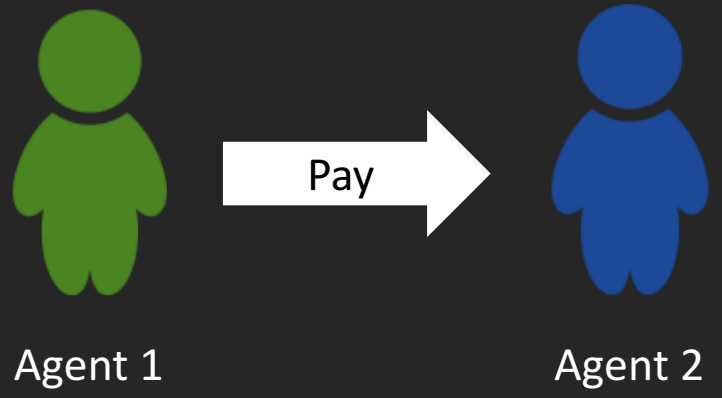
Serial number 1234567

+

Agent 1 PVk

+

Agent 1 send Agent 2 1 HSEcoin



Transactions log

We use a public-key encryption with digital signature

Serial numbers are auto generated by hash function

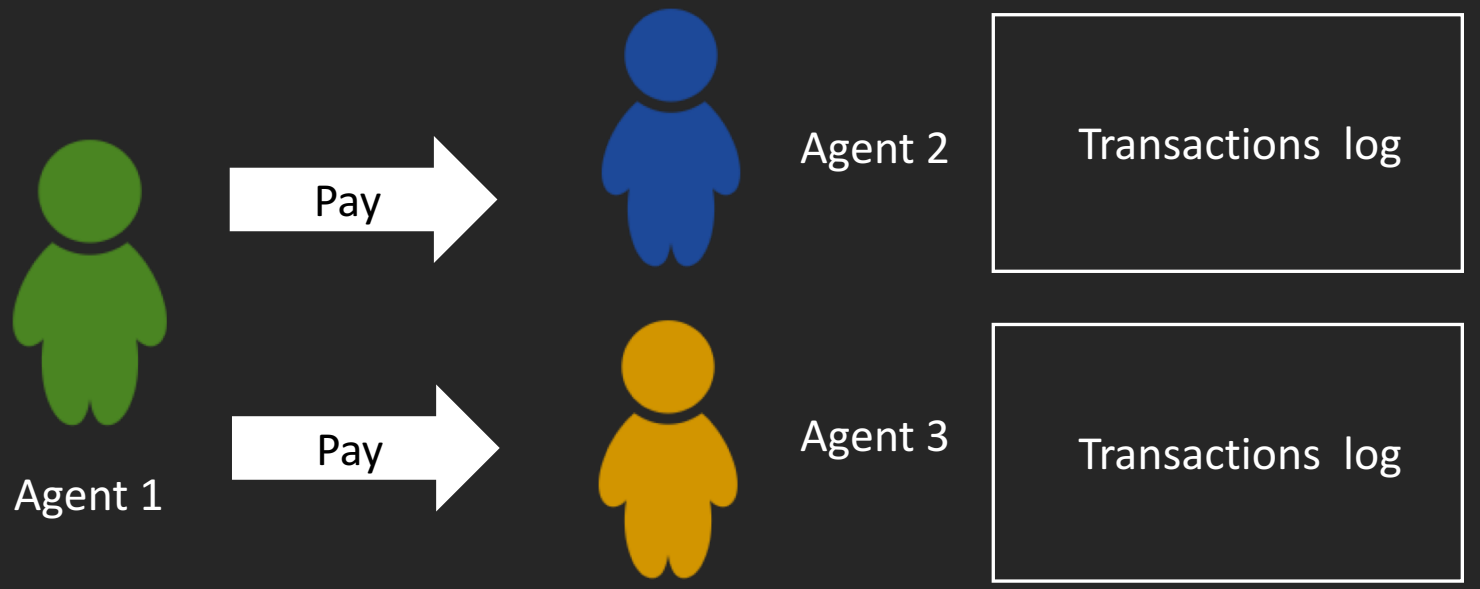
Serial number 1234567

+

Agent 1 PVk

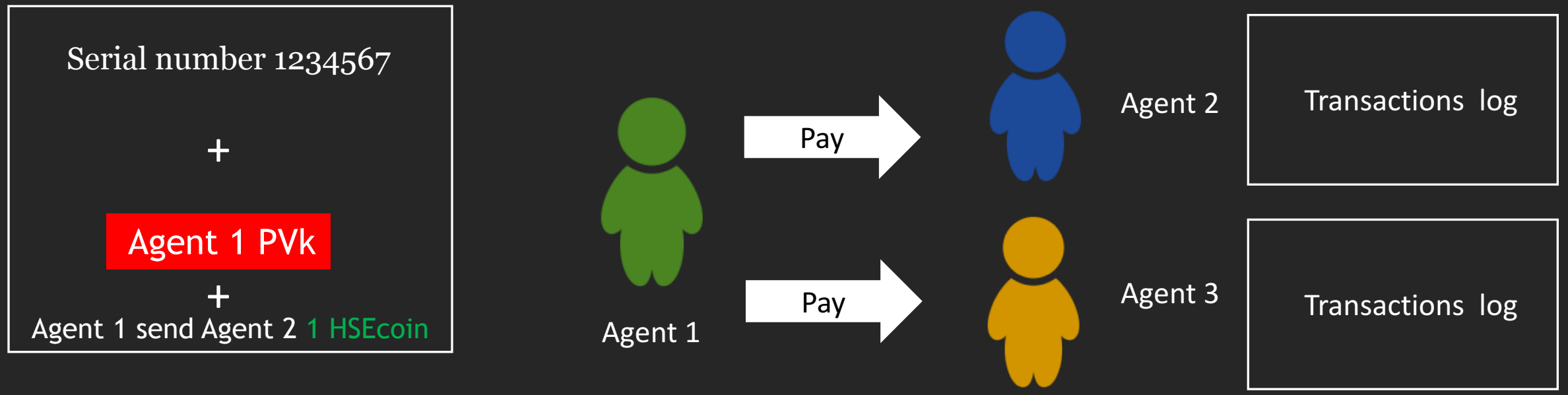
+

Agent 1 send Agent 2 1 HSEcoin



If Agent 1 will simultaneously send the same massage with the same serial number to Agent 2 and Agent 3, all of them confirm transaction

Double cost problem



All the world will be checking this transaction
If it will be “ok” agent 2 and agent 3 will accept it

if Agent 1 tries to spend his HSEcoin with both Agent 2 and Agent 3, other people on the network will notice, and network users will tell both Agent 2 and Agent 3 that there is a problem with the transaction, and the transaction shouldn’t go through

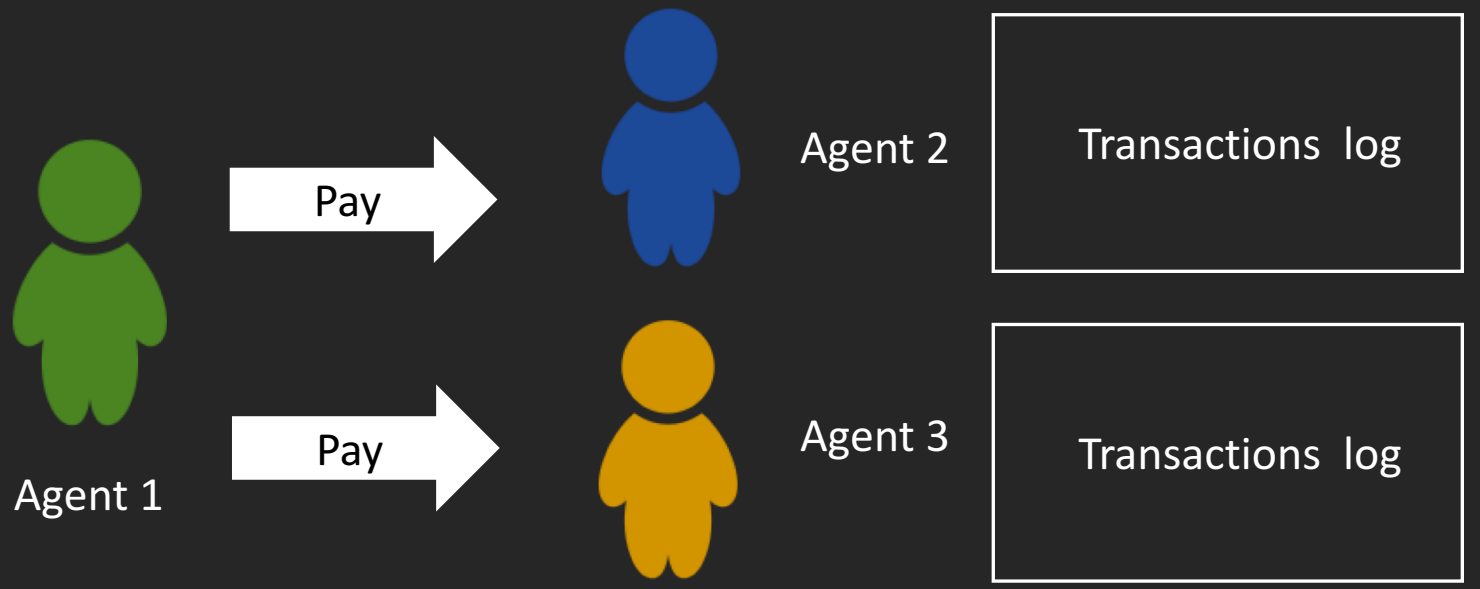
Serial number 1234567

+

Agent 1 PVk

+

Agent 1 send Agent 2 1 HSEcoin



If Agent 1 will simultaneously send the same massage with the same serial number to Agent 2 and Agent 3, all of them confirm transaction

How should other people update their block chains?

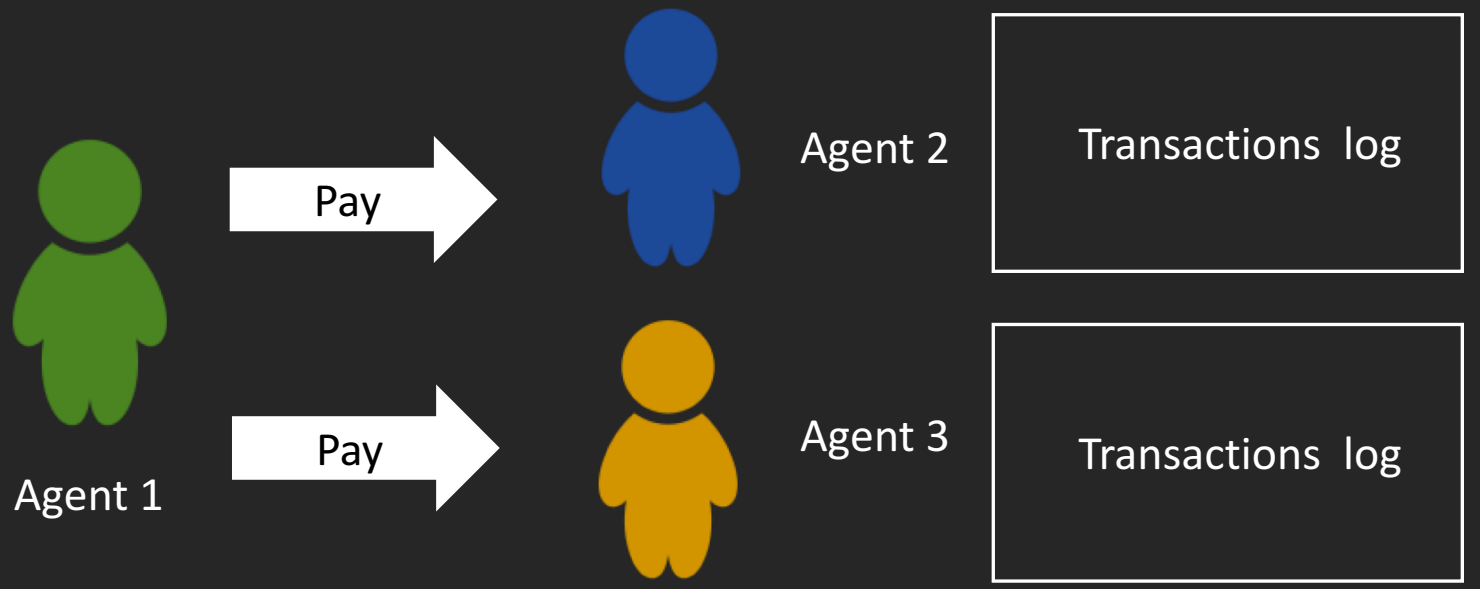
Serial number 1234567

+

Agent 1 PVk

+

Agent 1 send Agent 2 1 HSEcoin



Both of them will have their own transaction log

But

Agents 1 transaction will be verified after 3-6 new block

Serial number 1234567

+

Agent 1 PVk

+

Agent 1 send Agent 2 1 HSEcoin

Agent 1

Pay

Pay

Verify

Agent 2

Agent 3

Transactions log

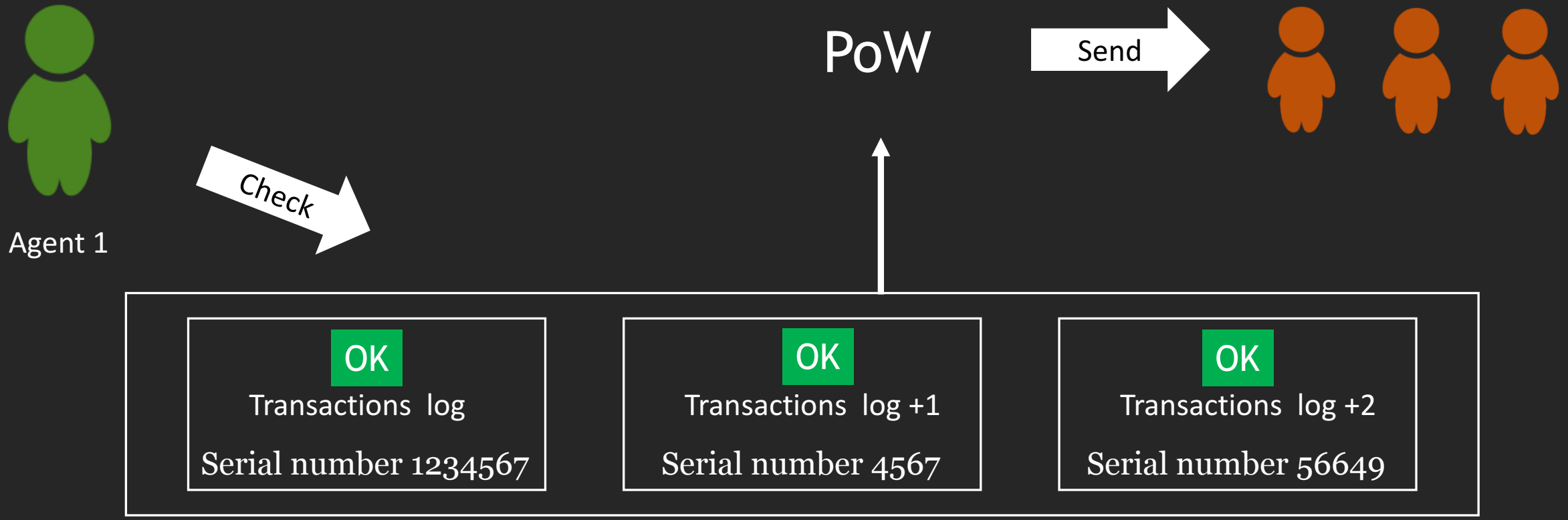
Transactions log

What if Agent 1 have a lot of friends who will help to make a double cost transaction?

Proof-of-work (PoW)

(1) To make it computationally costly for network users to validate transactions
(2) To reward them for trying to help validate transactions
(3) Validation can no longer be influenced by the number of network identities someone controls, but only by the total computational power

Proof-of-work (PoW)



Proof-of-work (PoW)

Let H be a fixed hash function
Let Agents 1 transaction with label J
Suppose Agent 1 appends a number X (called the nonce) to J and hashes the combination

Proof-of-work (PoW)

“hello world!” = J and X = “0”
We have :

H("Hello, world!0")=1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64

Find X such that when we append X to J
We will have:

H("JX")=

0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9

Proof-of-work (PoW)

In the Bitcoin protocol, this validation process is called mining

HSEcoin

1. Decentralized registry with transaction log
2. Every person have the same registry
3. PoW verification

1. Decentralized registry with transaction log
2. Every person have the same registry
3. PoW verification

CAP theorem

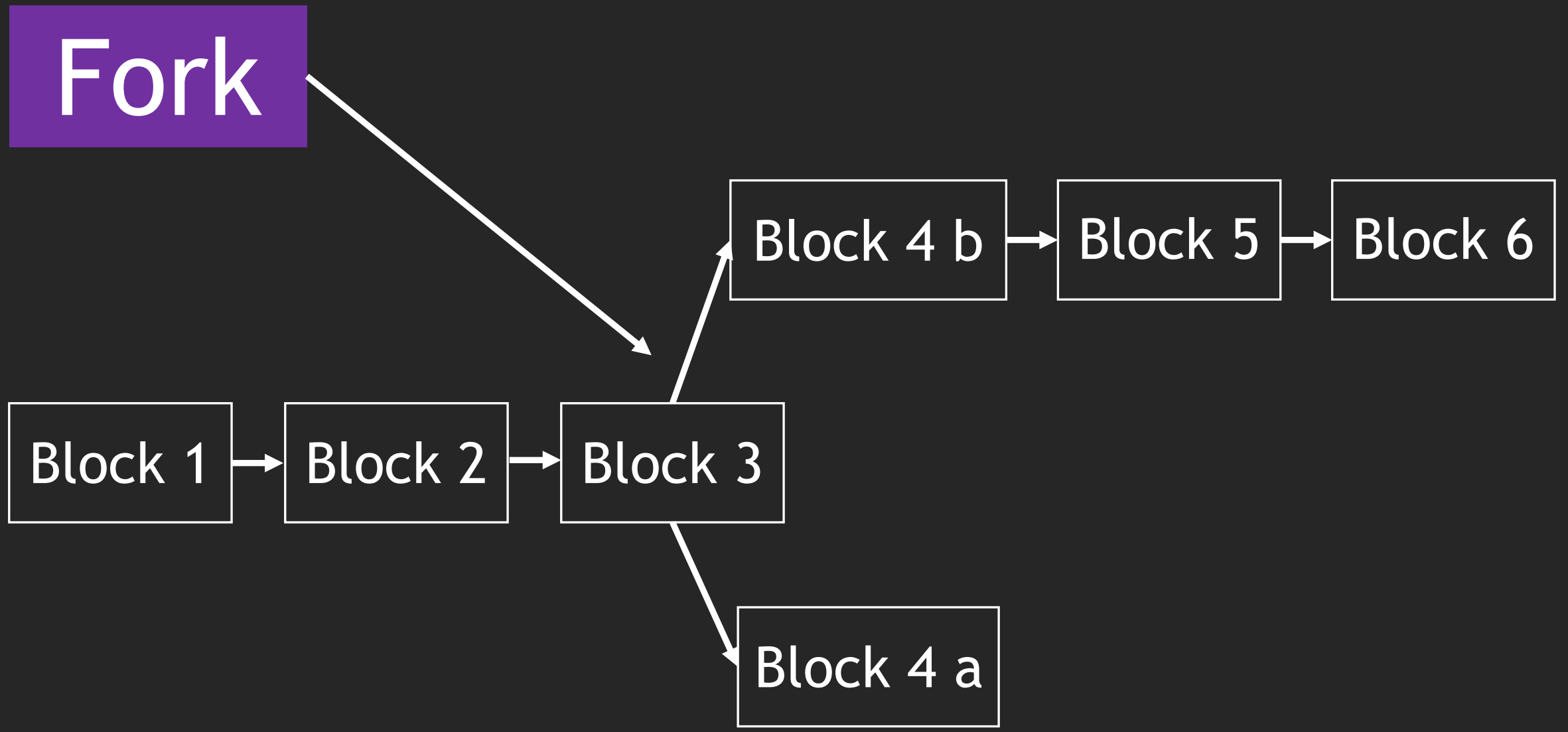
CAP - Consistency, Availability, Partition tolerance. Can’t have all three!

If a network partition takes place you can’t have availability and consistency

FLP impossibility(Fisher, Lynch, Patterson)

No*de deterministic* algorithm can guarantee consensus in asynchronous setting

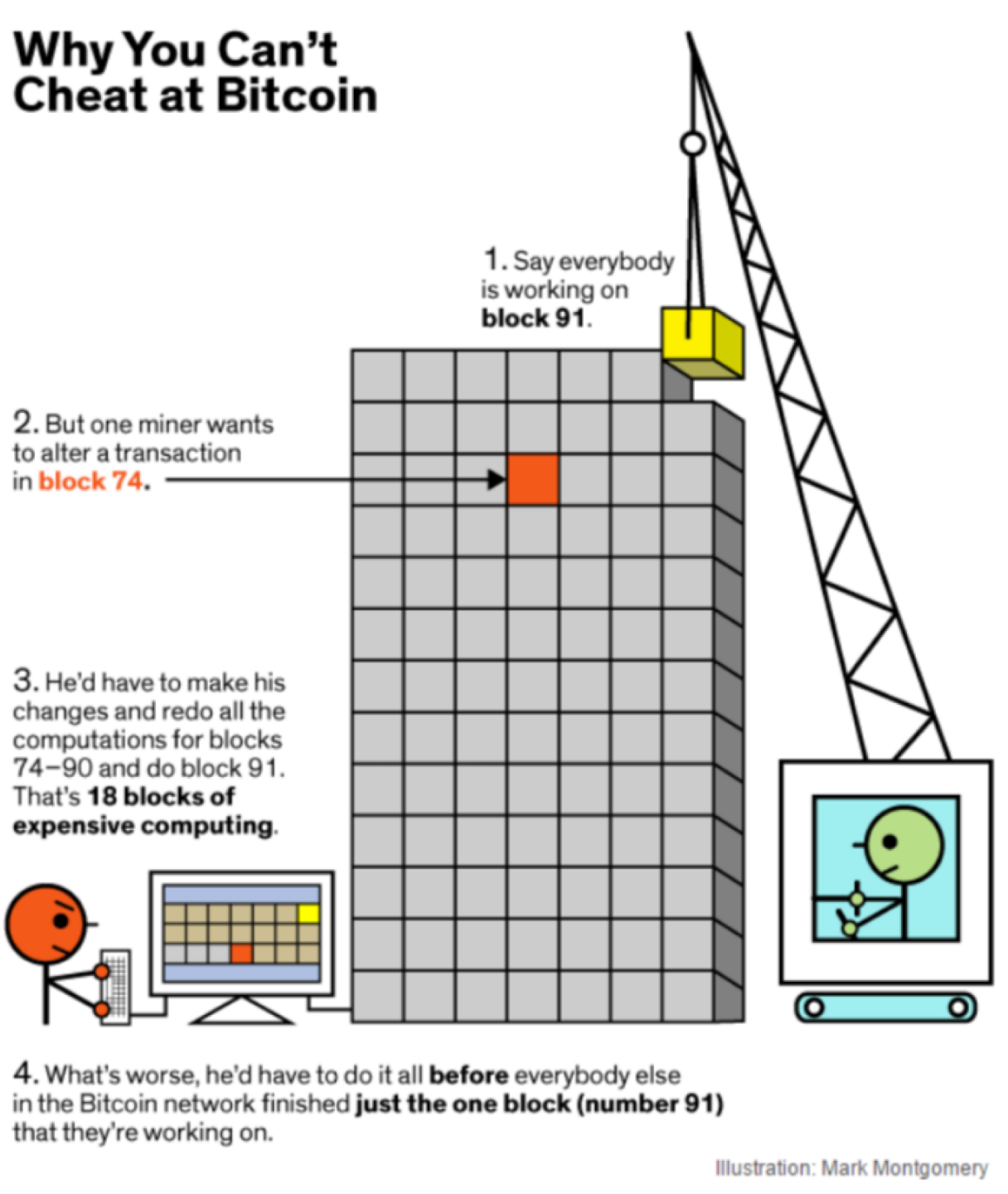
Dwork, Lynch, Stockmeyer 1984 Bounds on fault tolerance


Fork
Block 4 b
Block 5
Block 6
Block 1
Block 2
Block 3
Block 4 a

If Hash Block 4 b is longer than Block 4 a

All miners will go to Block 4 b

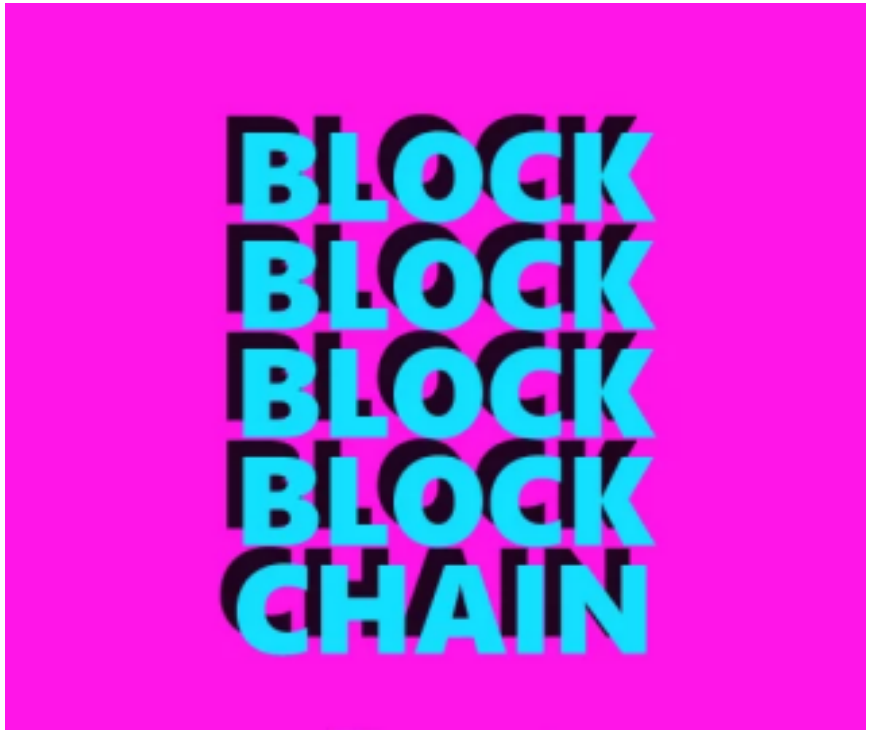
All the transactions will be verified only after 6 blocks



Questions

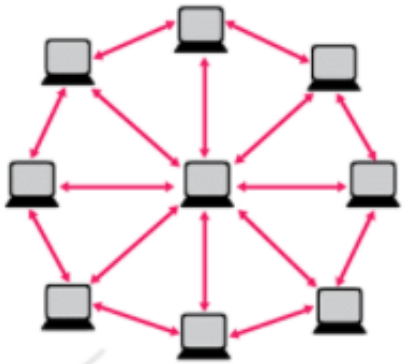
Ethereum

The blockchain technology



The purpose

Someone requests a transaction.



The requested transaction is broadcast to a **P2P network consisting of computers, known as nodes.**

Validation

The network of nodes **validates the transaction and the user's status using known algorithms.**

A verified transaction can involve **cryptocurrency,** contracts, records, or other information.

cryptocurrency

Once verified, the transaction is combined with other transactions **to create a new block of data for the ledger.**

The new block is then added to the existing blockchain, in a way that is permanent and unalterable.

The transaction is complete.

Has no intrinsic value in that is not redeemable for another commodity such as gold.

Has no physical form and exists only in the network.

Its supply is not determined by a central bank and the network is completely decentralized.

The creator

"I have the chance to help build software and tools that already affect tens of thousands of people around the world, and to work on advanced problems in computer science, economics and philosophy every week."

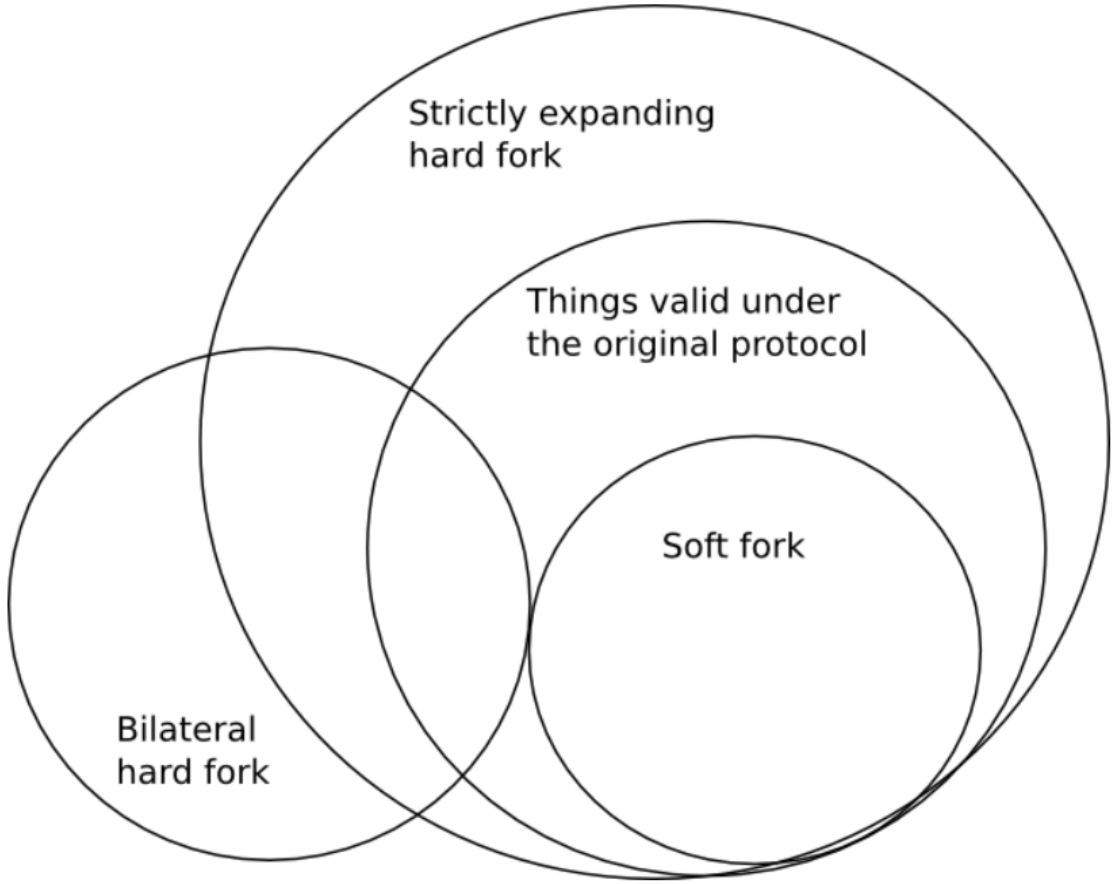
Vitalik Buterin

ETHEREUM

- Public
- Open-source
- Features smart contract (scripting) functionality
- Blockchain-based distributed computing platform

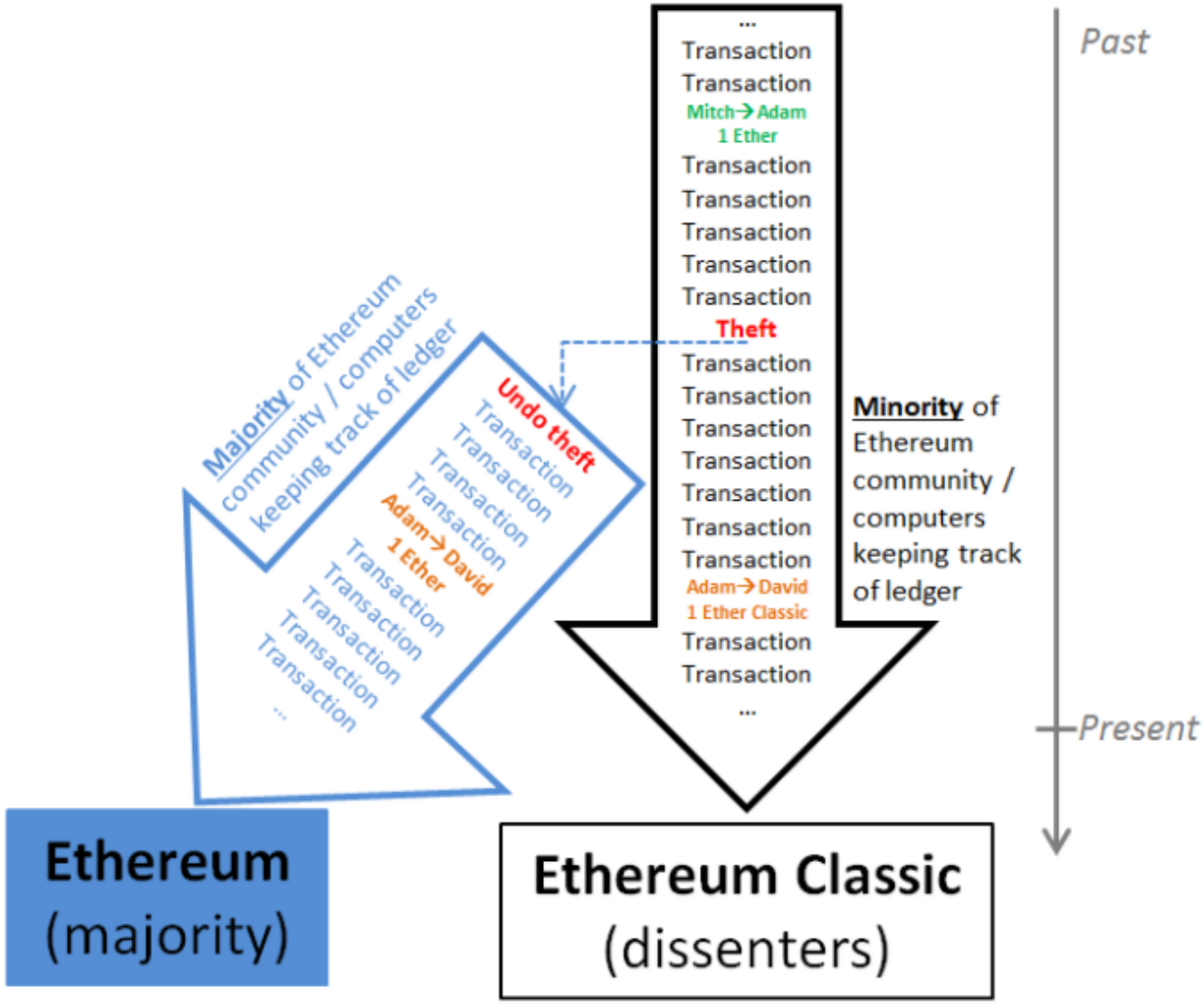
The Ethereum fork

Can we change blockchain?

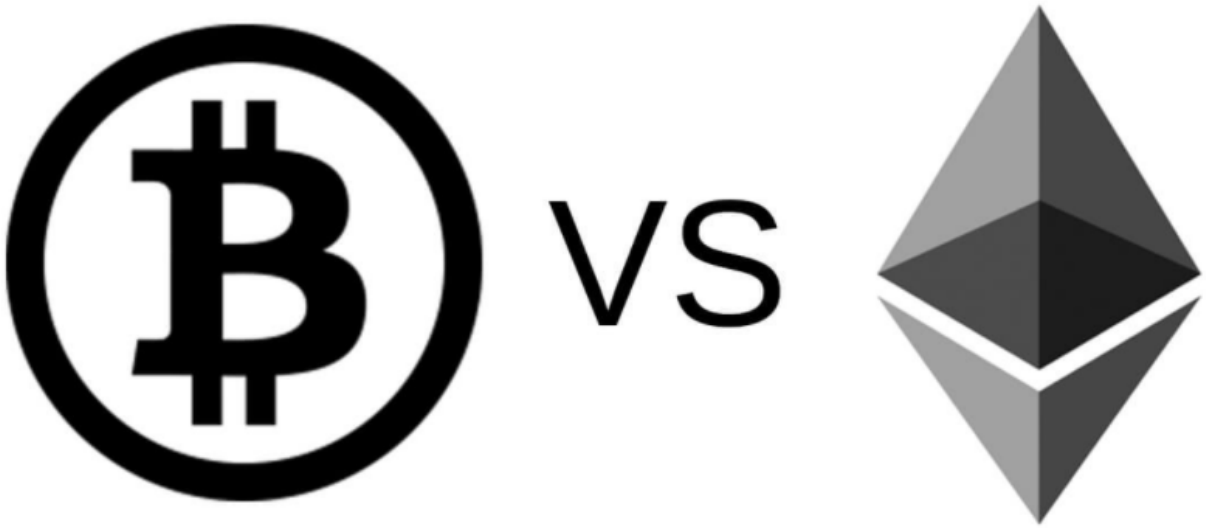


Can we change blockchain?

A soft fork where in it's your choice whether you want to update or not.



They are very different

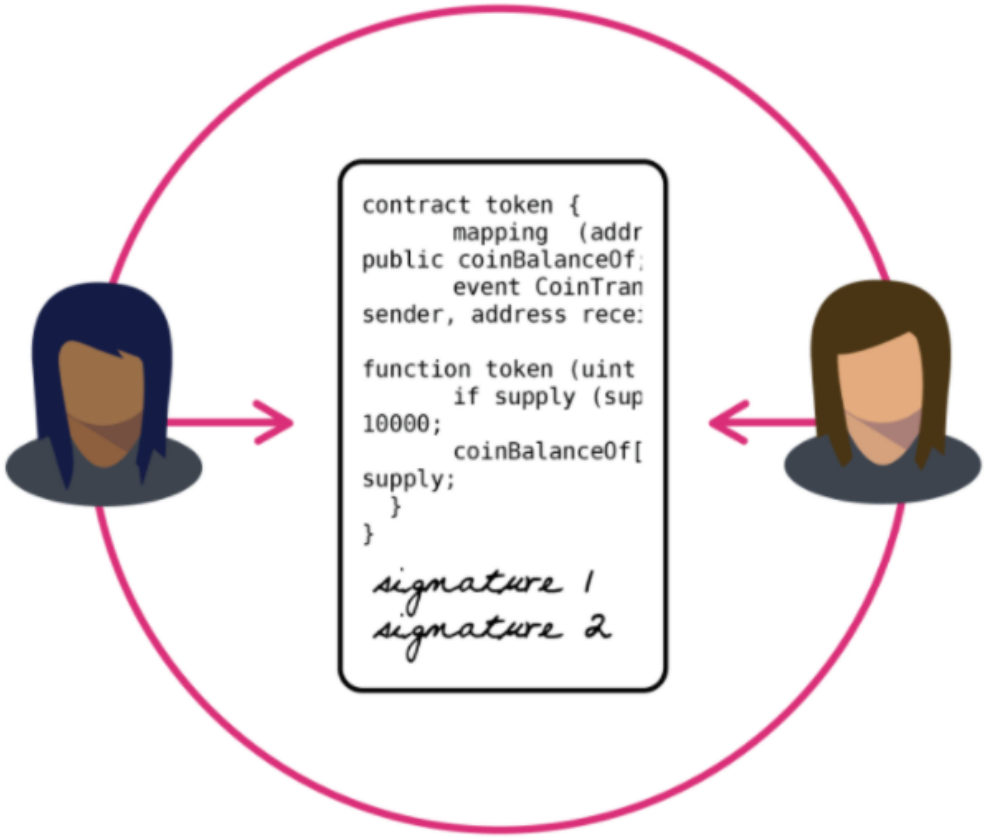


How can I use it?

- Gambling (probably fair)
- Storage for any form of contract document that none of the sides can alter
- Storage for data that can never be lost
- Fair and transparent auctions
- All sorts of smart voting mechanisms
- Raising money

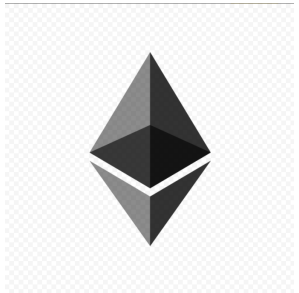
Smart contract

A computer protocol intended to facilitate, verify, or enforce the negotiation or performance of a contract.



Provenance

Enigma



Ethereum

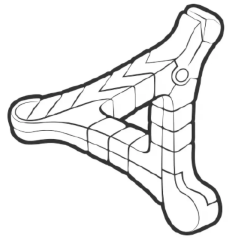
Consensus

Ethcore

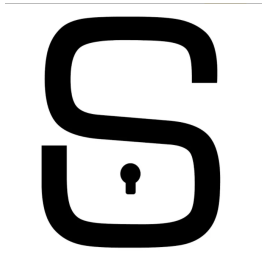
IPFS

Colony

Plex



Backfeed

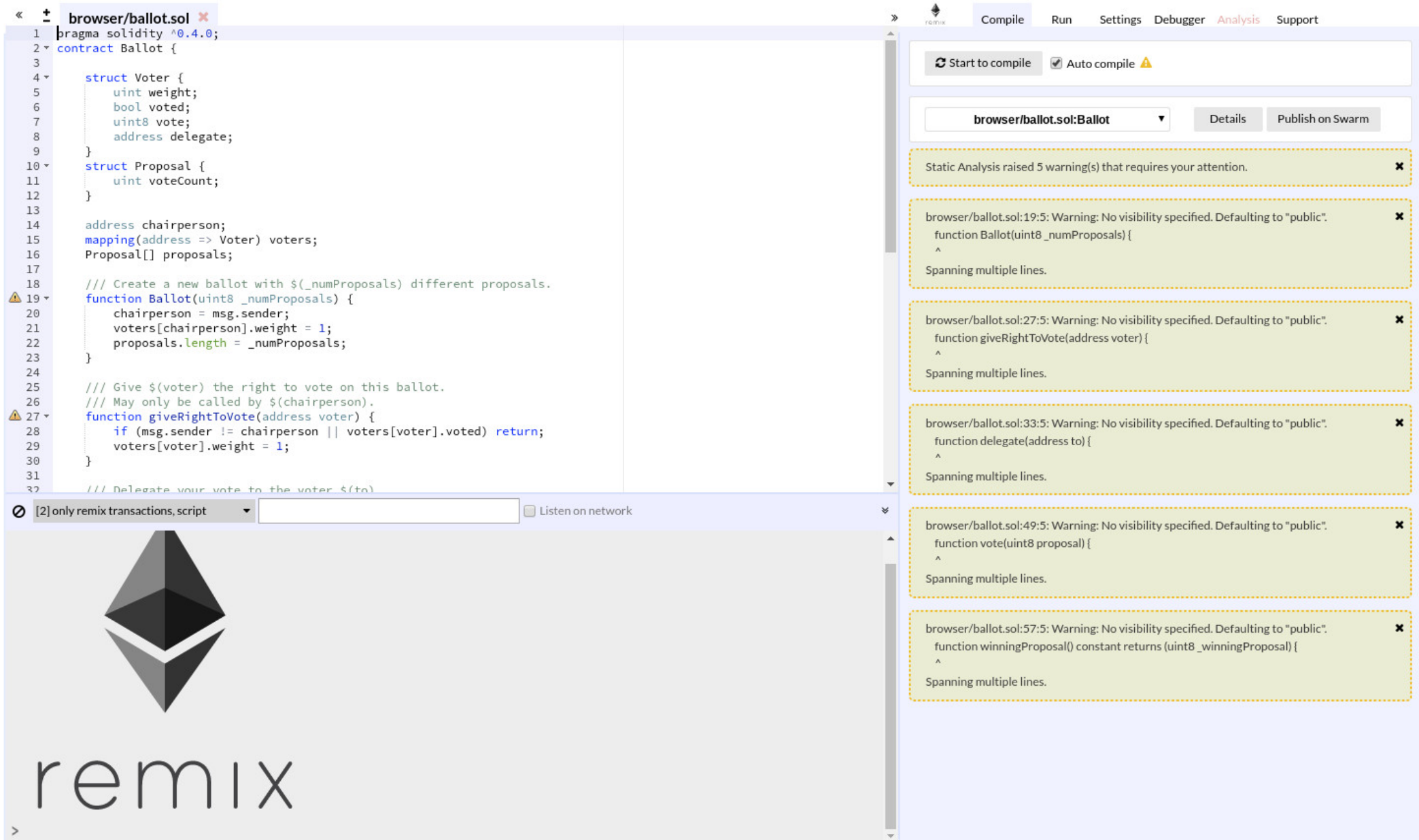


SlockIt

What do I need to use it?

- Solidity
- Remix
- Internet
- Desire

- ICO



Is Ethereum the future?

Smart contracts

- Provide security that is superior to traditional contract law and to reduce other transaction costs associated with contracting
- New ways to formalize the relationships
- Contain code functions and can interact with other contracts, make decisions, store data, and send ether to others
- Programming language = Solidity

Smart contracts


```
contract Option {
        trikePrice = $50
        holder = Alice
        seller = Bob
        asset = 100 shares of Acme Inc.
        expiryDate = June 1st, 2016
function exercise ( ) {
        If Message Sender = holder, and
        If Current Date<expiryDate, then
                holder send($5,000) to seller, and
                seller send(asset) to holder
                }}
```

DApp

DApp

1. DApp is smart contracts evolution from automatization point of view
2. Dapp – is group of smart-contracts in the Backend with GUI in the Frontend
3. We can make their implementation by uploading code into transaction and pasting it into our network

DApp

OpenBazaar

LaZooz

Twister

Gems

Storj

Our plans

Next time

- Blockchain on private network
- Business cases
- Ethereum smart contract practice

A bit later

- Security and future aspects
- Blockchain + Azure
- Ethereum Consortium Leader

Resources

- Melanie Swan - “Blockchain”
- http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works
- https://ethereum.org

Thank You!

Nick Kozlov - MSP (Rus)
Nikita.Kozlov@studentpartner.com

Dasha Slesareva
sls-daria99@yandex.ru