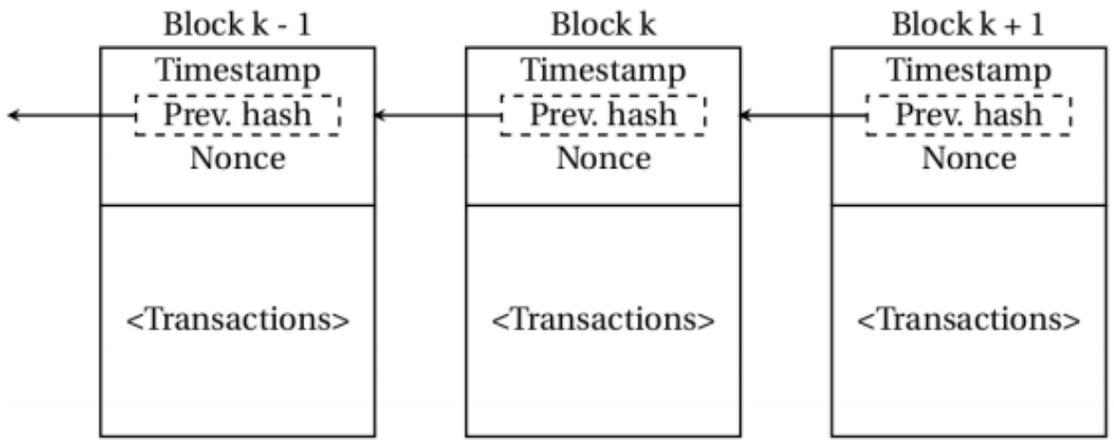
Security Issues in Swarm Robotics



Dr. Volker Strobel

Postdoctoral researcher

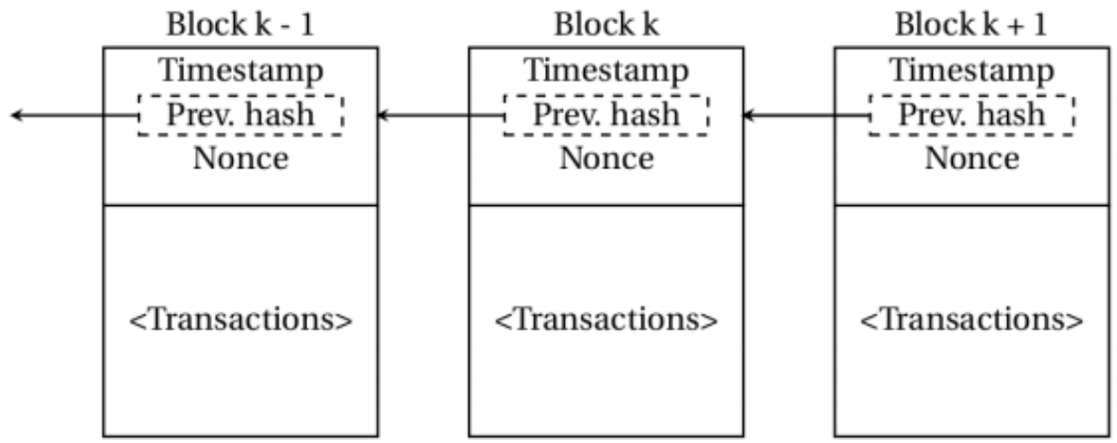
IRIDIA

ULB

ECOLE POLYTECHNIQUE DE BRUXELLES

Security Issues in Swarm Robotics

with a focus on blockchain technology



Dr. Volker Strobel

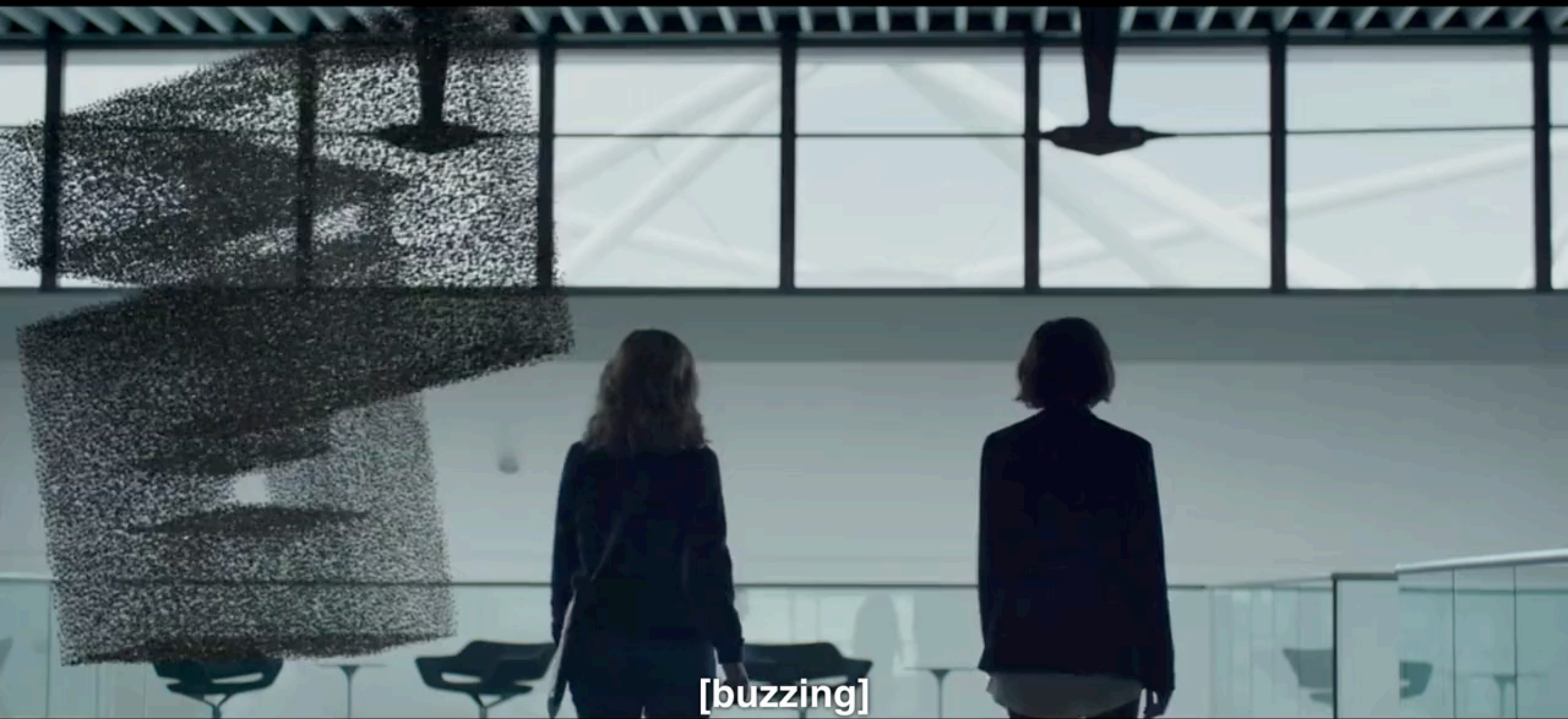
Postdoctoral researcher

16 May 2024

IRIDIA

ULB

ECOLE POLYTECHNIQUE DE BRUXELLES

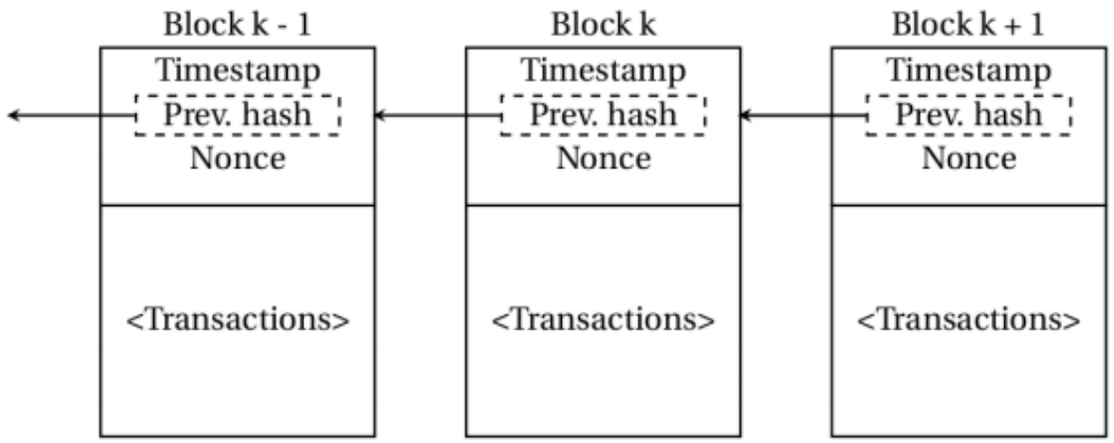


Black Mirror - Season 3 Episode 6 - Hated in the Nation

Spoiler alert!


You're scaring the shit out of me!

Security Issues in Swarm Robotics



Dr. Volker Strobel

Postdoctoral researcher

IRIDIA

ULB

ECOLE POLYTECHNIQUE DE BRUXELLES

Security issues in swarm robotics

Part I
Shortcomings of classical approaches in swarm robotics

Part II
Blockchain technology

Part III
Blockchain-based smart contracts for the secure coordination of robot swarms

Security issues in swarm robotics

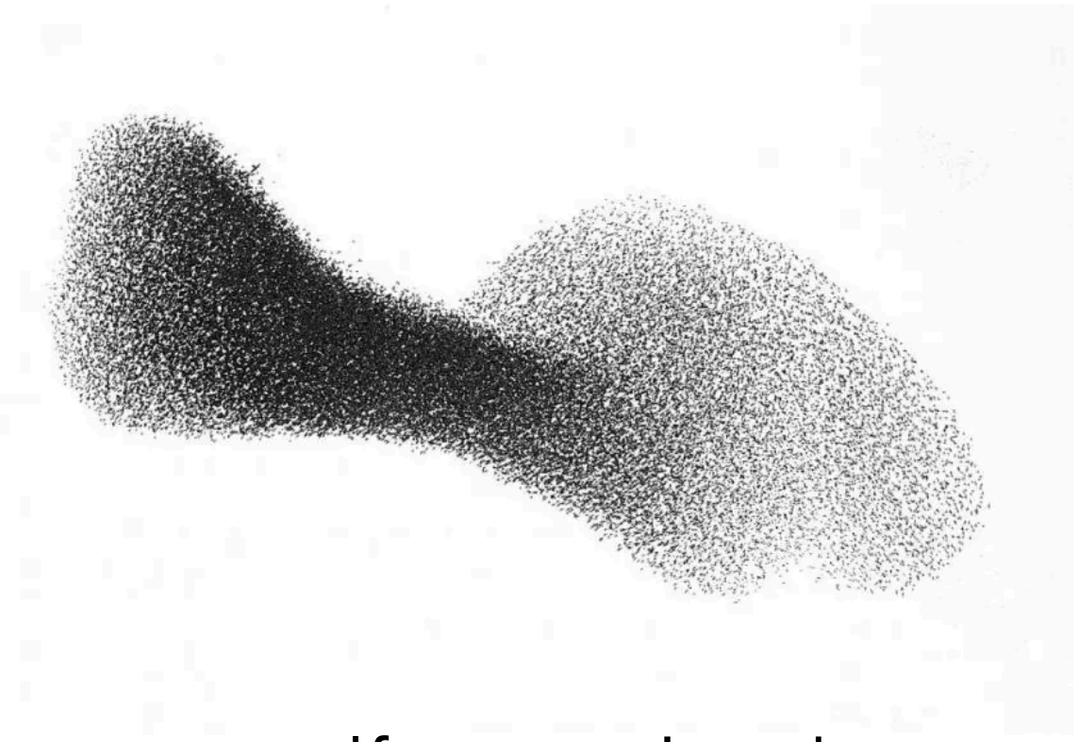
Part I
Shortcomings of classical approaches in swarm robotics

Part II
Blockchain technology

Part III
Blockchain-based smart contracts for the secure coordination of robot swarms

Fundamentals of swarm robotics

inspired by
swarm intelligence



self-organized

decentralized

Why no central control?

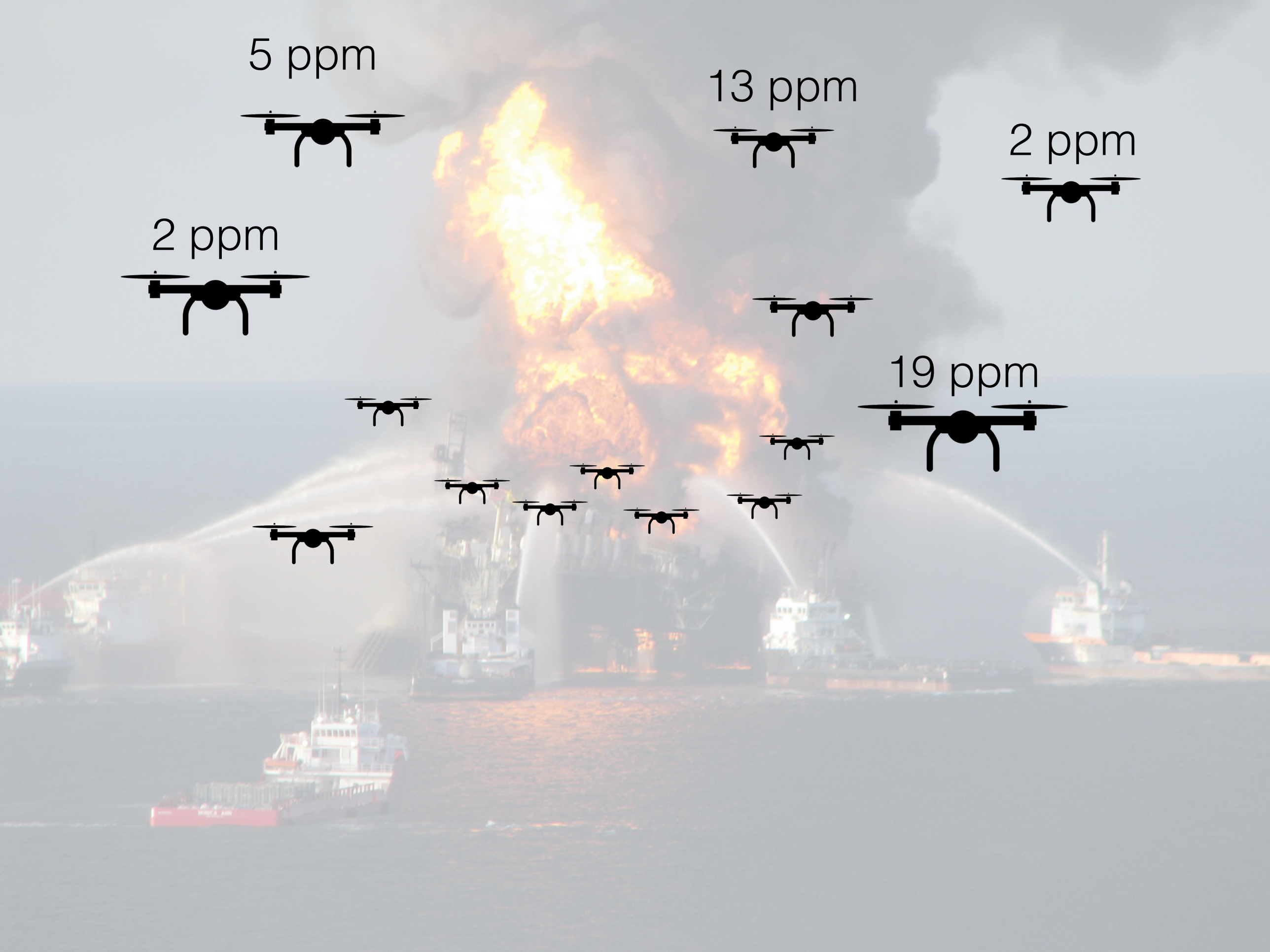
Decentralization:

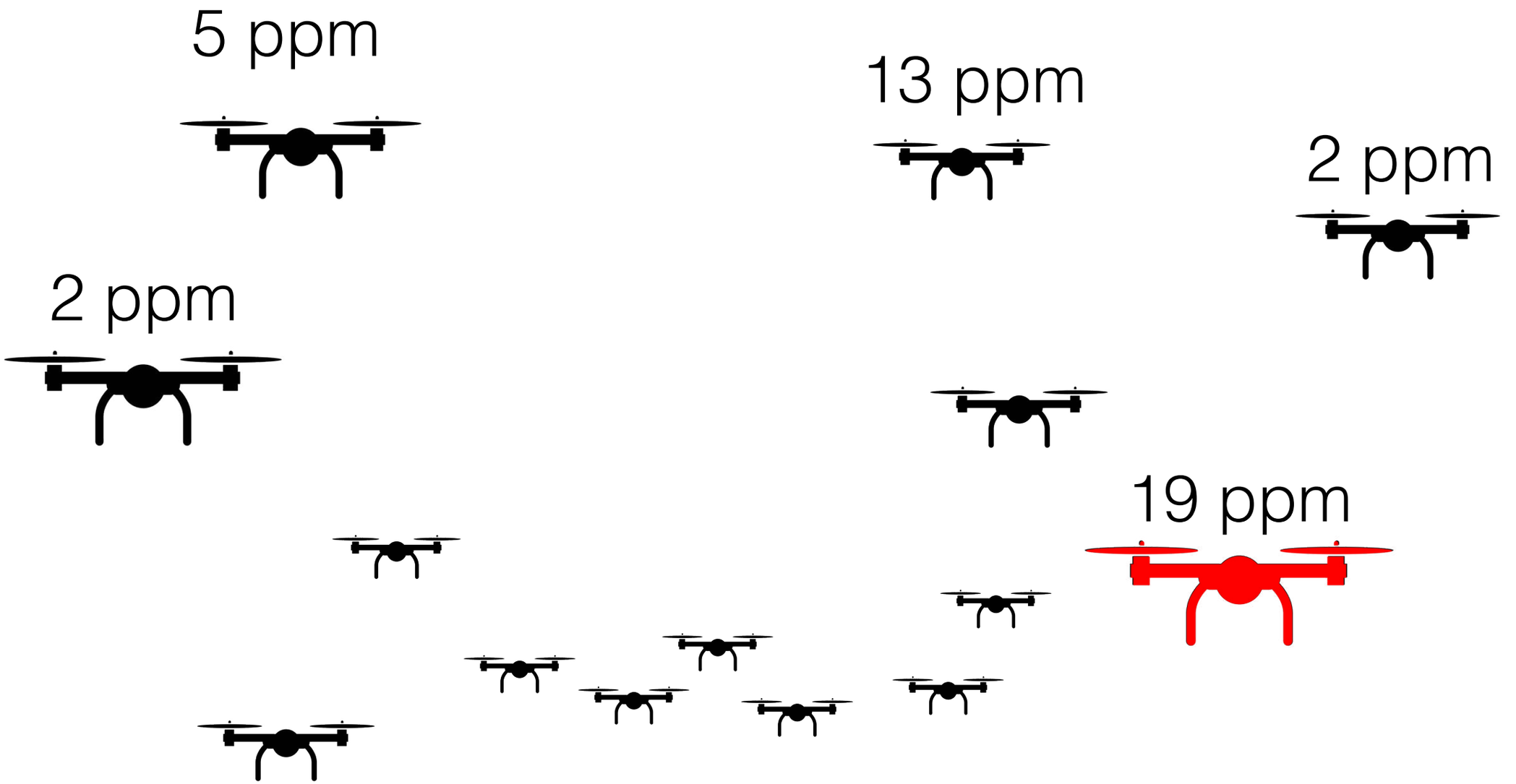
- prevents a single point of failure
- leads to less communication overhead
- enables remote deployments

Why no central control?

Decentralization:

- prevents a single point of failure
- leads to less communication overhead
- enables remote deployments


5 ppm
13 ppm
2 ppm
2 ppm
19 ppm


5 ppm
13 ppm
2 ppm
2 ppm
19 ppm

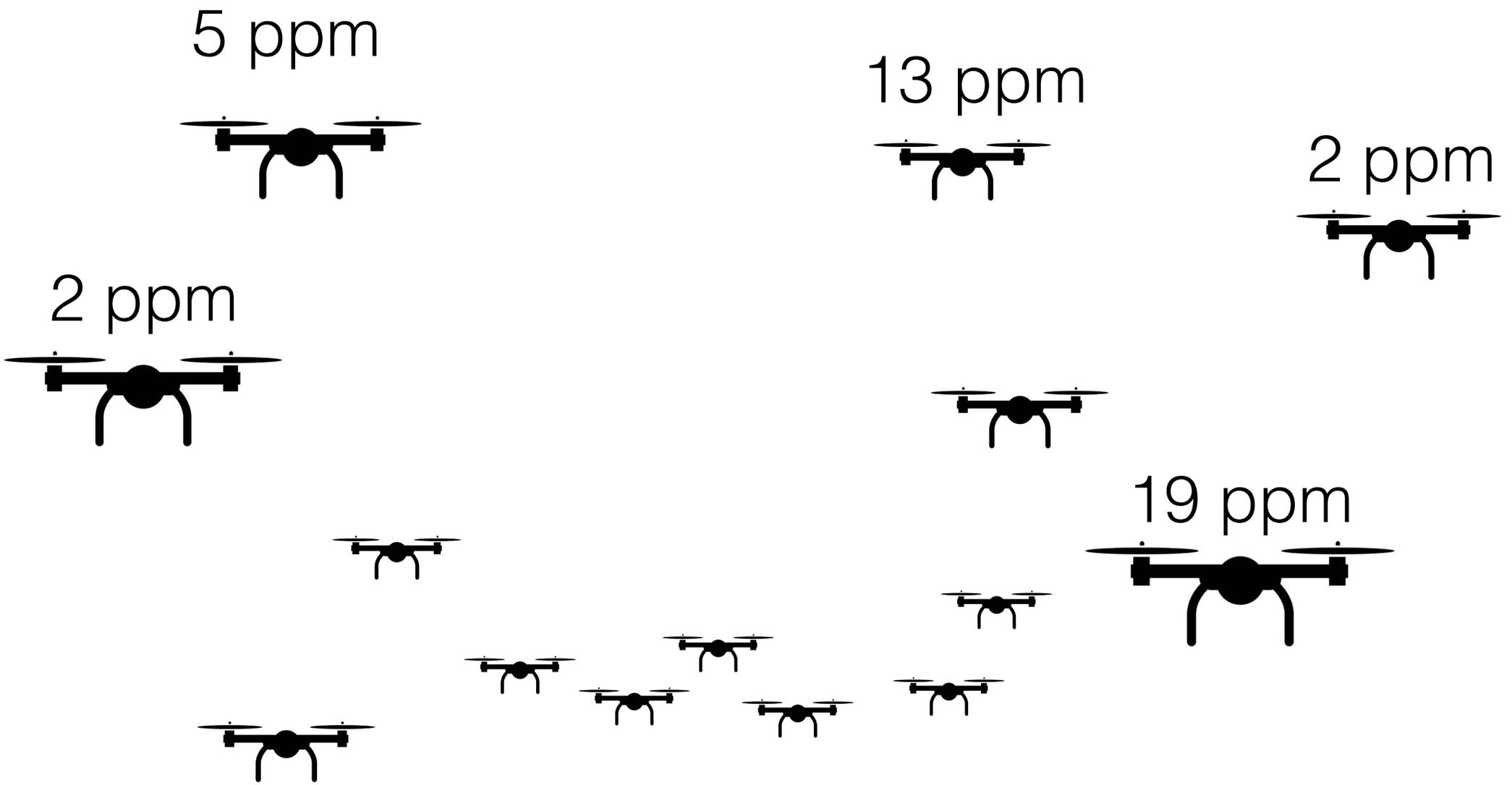
5 ppm

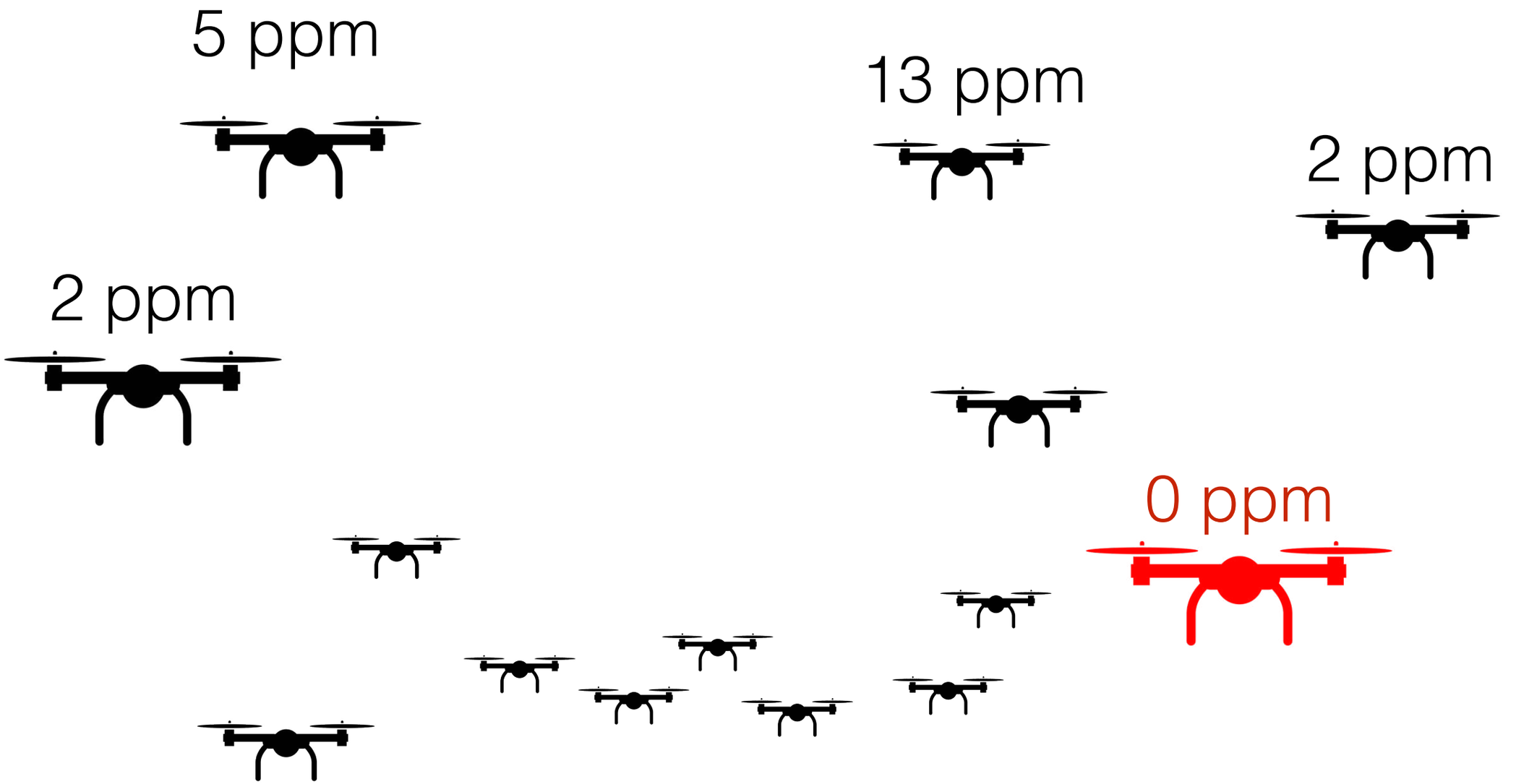
13 ppm

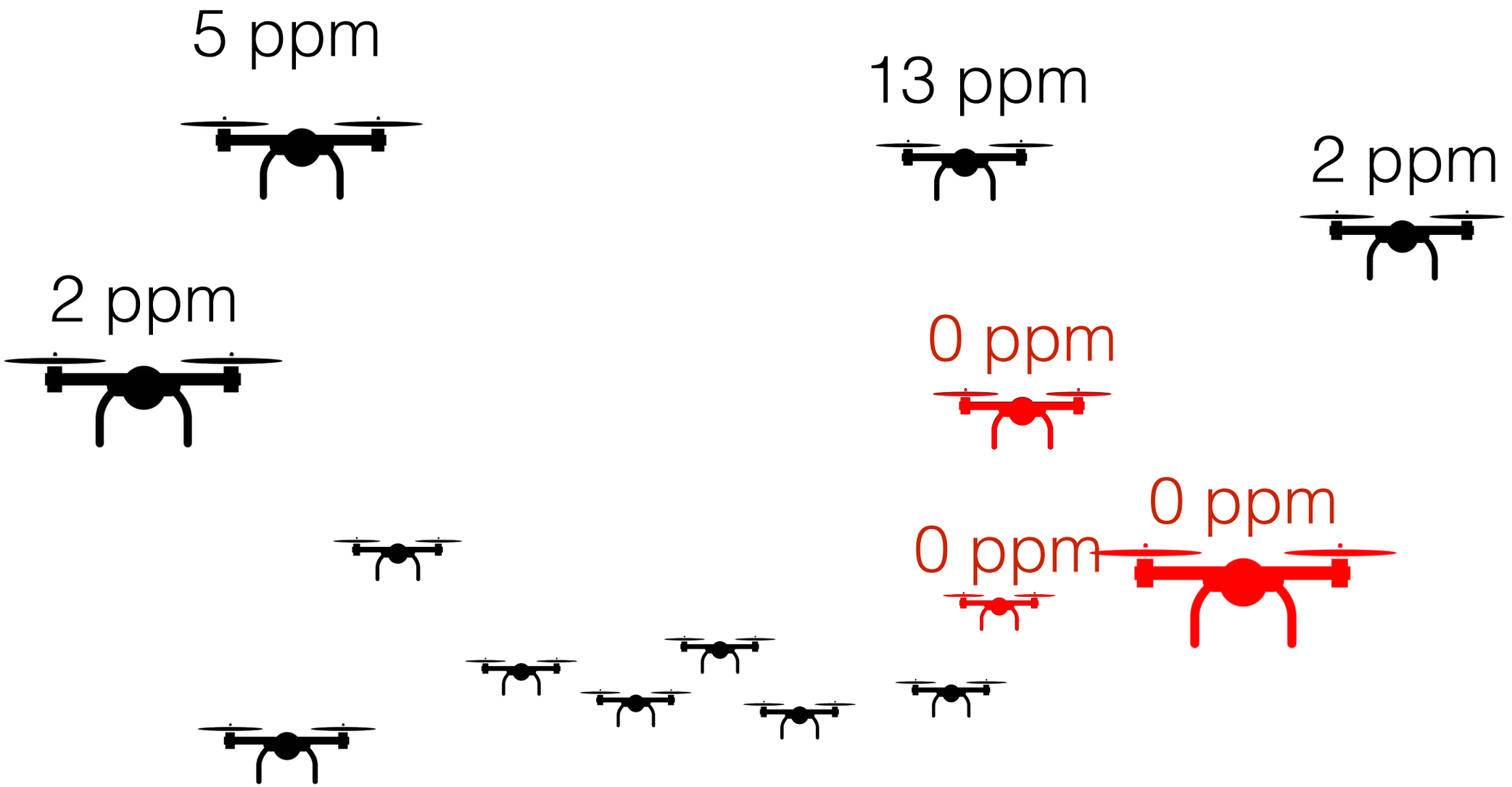
2 ppm

2 ppm

Fault tolerance by redundancy


5 ppm
13 ppm
2 ppm
2 ppm
19 ppm


5 ppm
13 ppm
2 ppm
2 ppm
0 ppm


5 ppm
13 ppm
2 ppm
2 ppm
0 ppm
0 ppm
0 ppm

5 ppm

13 ppm

2 ppm

2 ppm

0 ppm

0 ppm

0 ppm

0 ppm

5 ppm

13 ppm

2 ppm

2 ppm

0 ppm

0 ppm

0 ppm

0 ppm

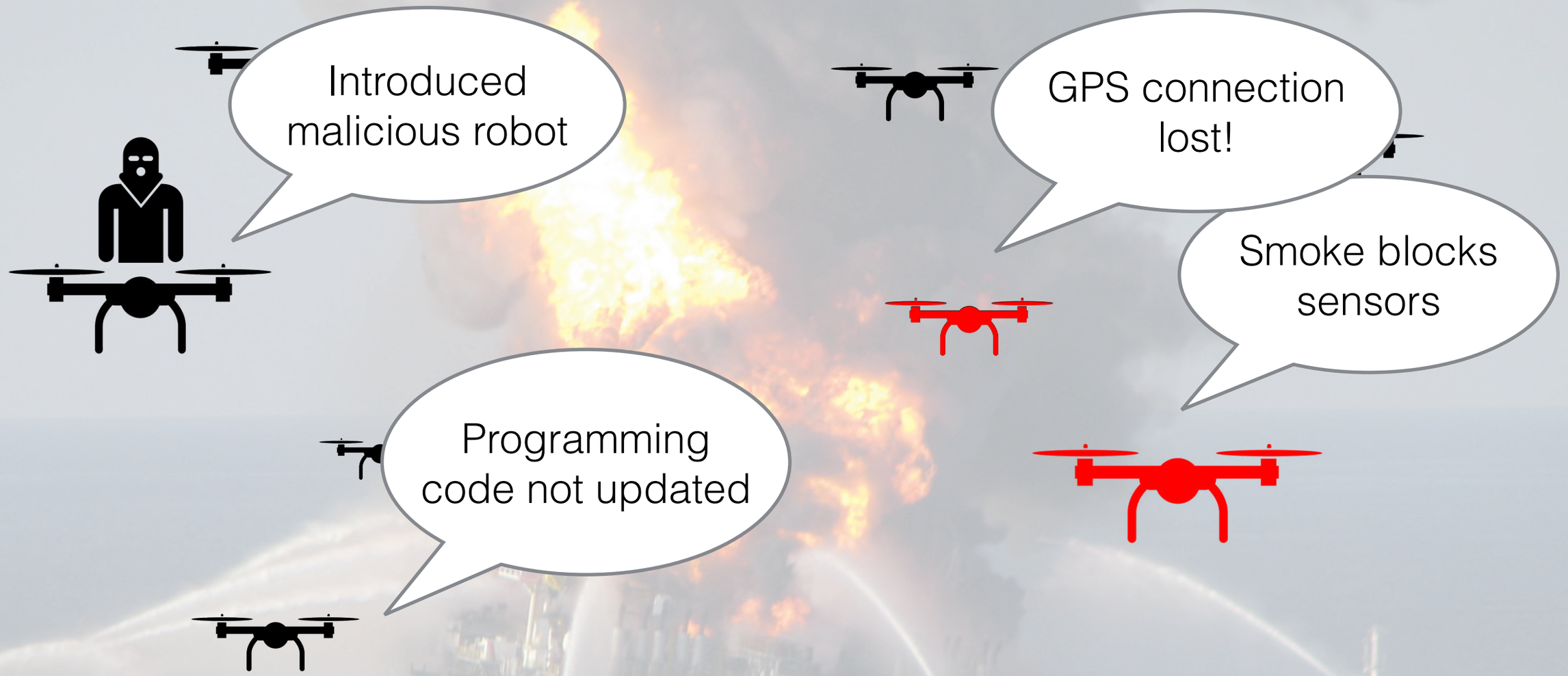
0 ppm

0 ppm

0 ppm

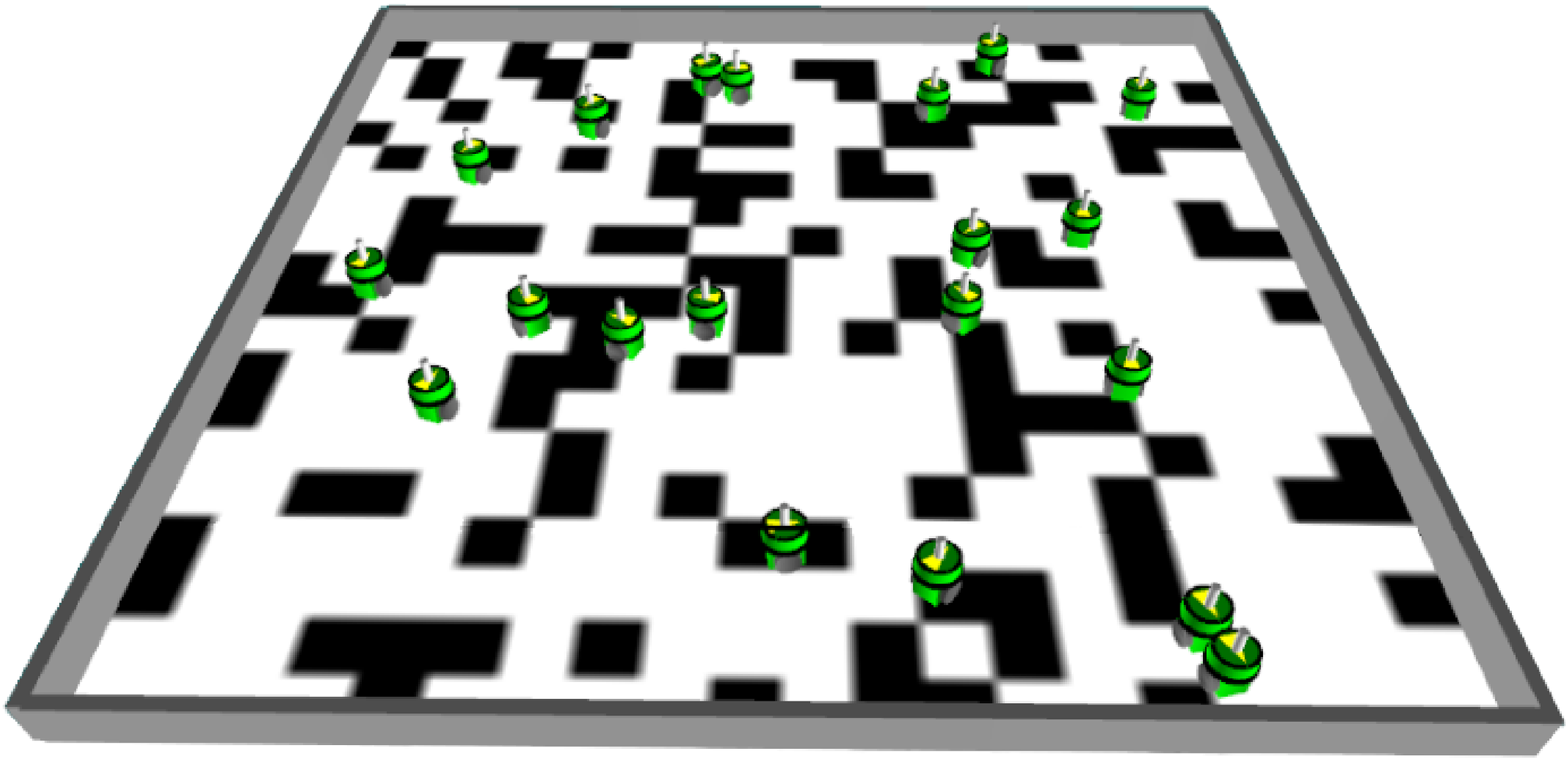
Byzantine robots show faulty or malicious behavior

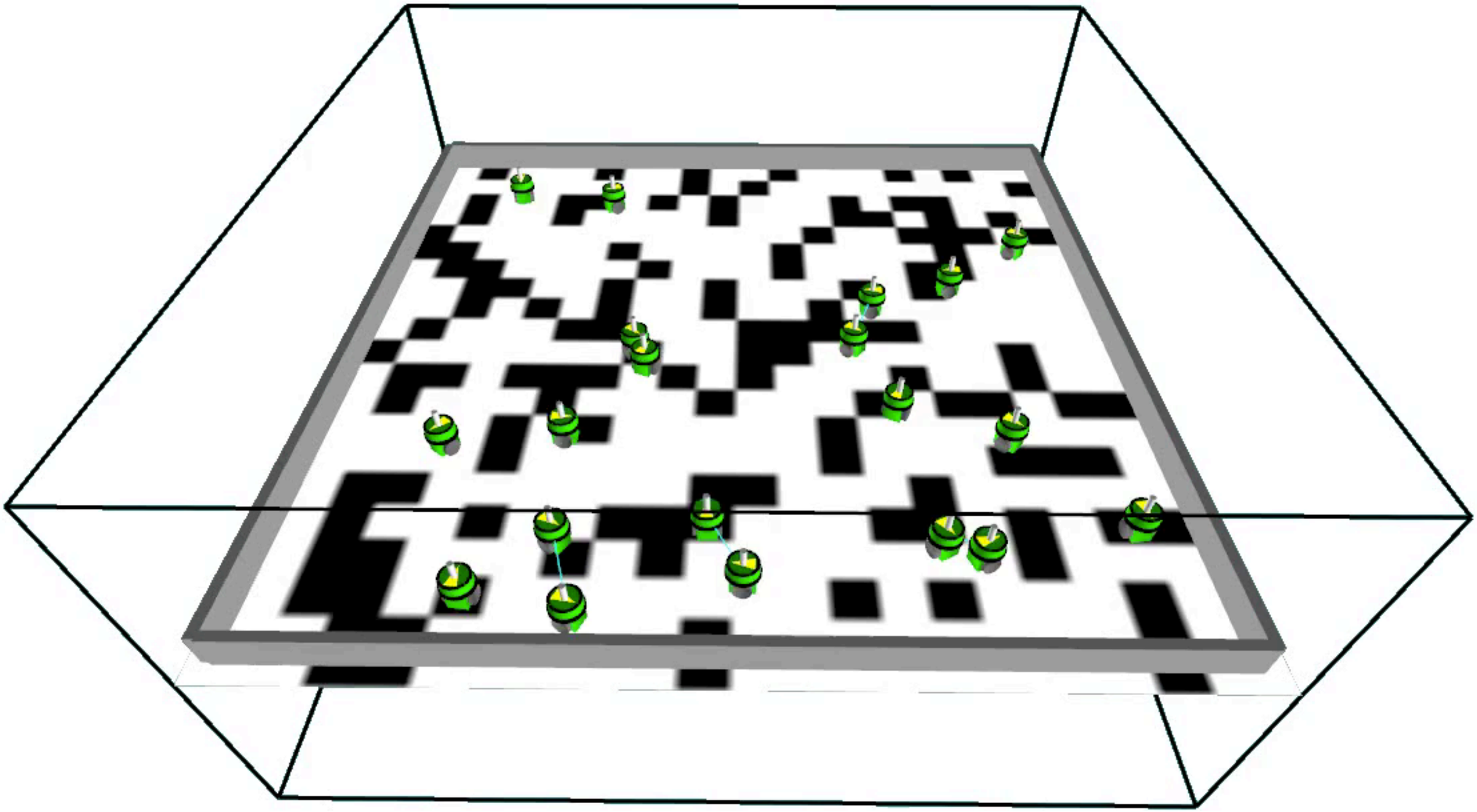
Intended behavior		**Actual behavior**
13 ppm 		0 ppm  **Byzantine**



Byzantine robots show faulty or malicious behavior

Continuous decision-making problem:
The robots' task is to determine the **fraction** of white tiles





Byzantine linear approximate consensus (classical approach)

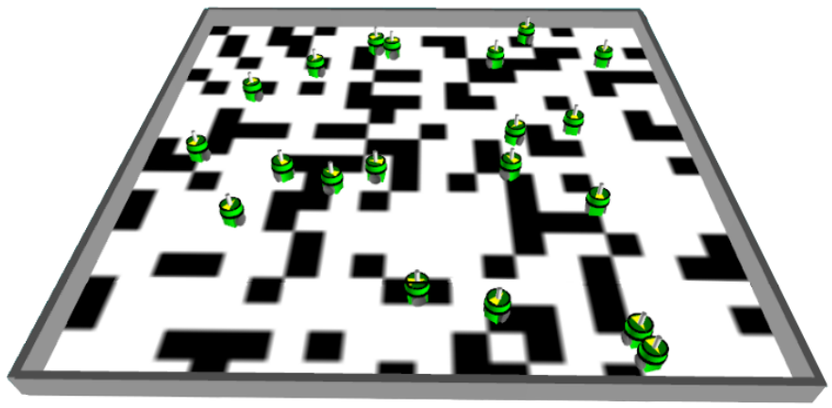
The algorithm calculates a
weighted sum between a robot's own estimate and its neighbors

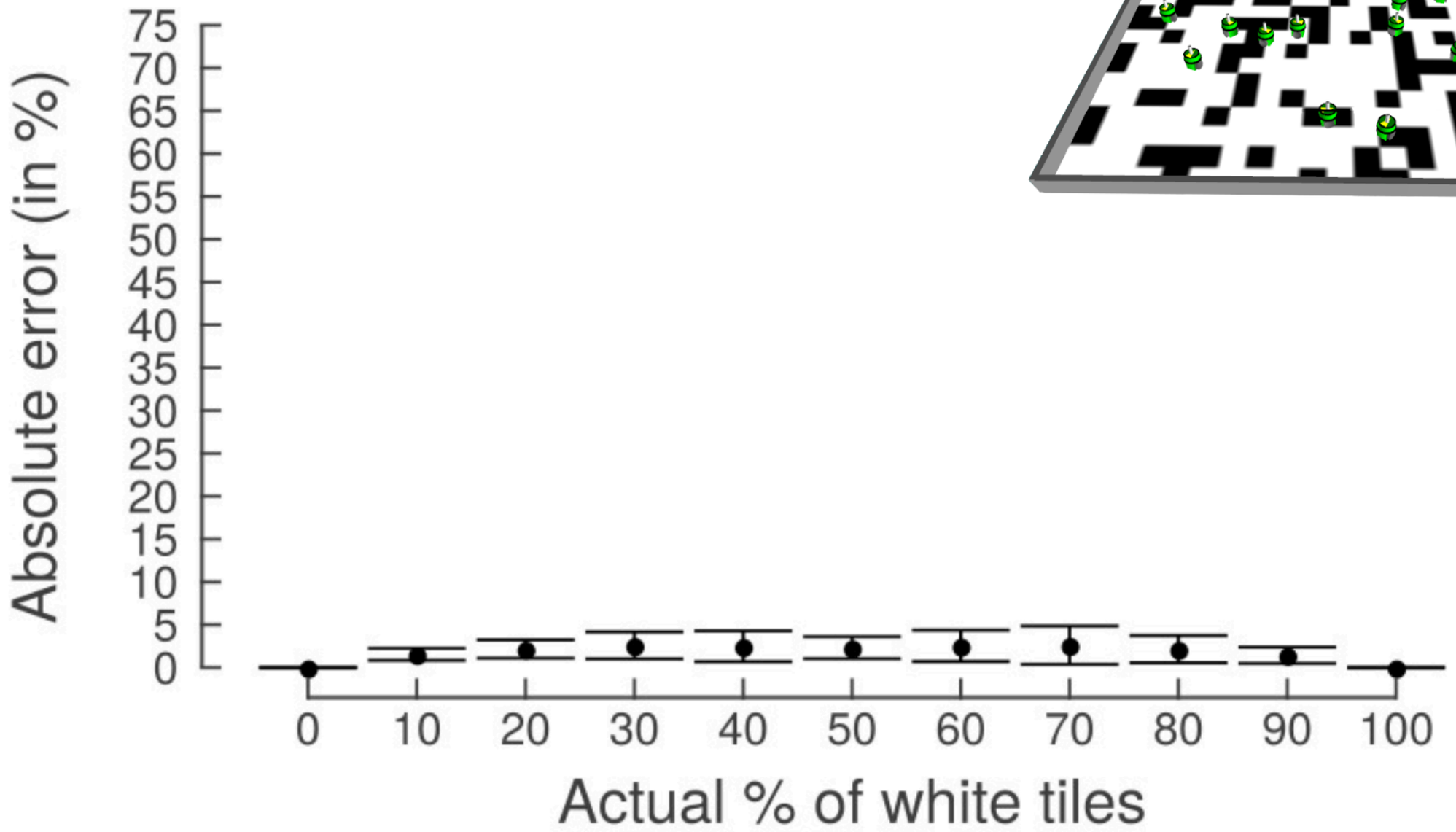
$$h_i(t) = x_i(t) + \sum_{j \in N_i} x_j(t)$$

- $h_i(t)$: robot's estimate
- $x_i(t)$: robot's own sensor reading
- $x_j(t)$: value disseminated by neighbor

discard largest and smallest values

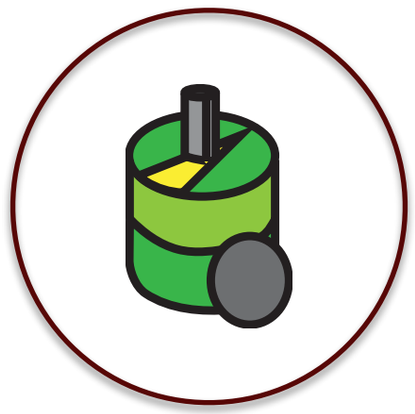
The algorithm is able to achieve the goal





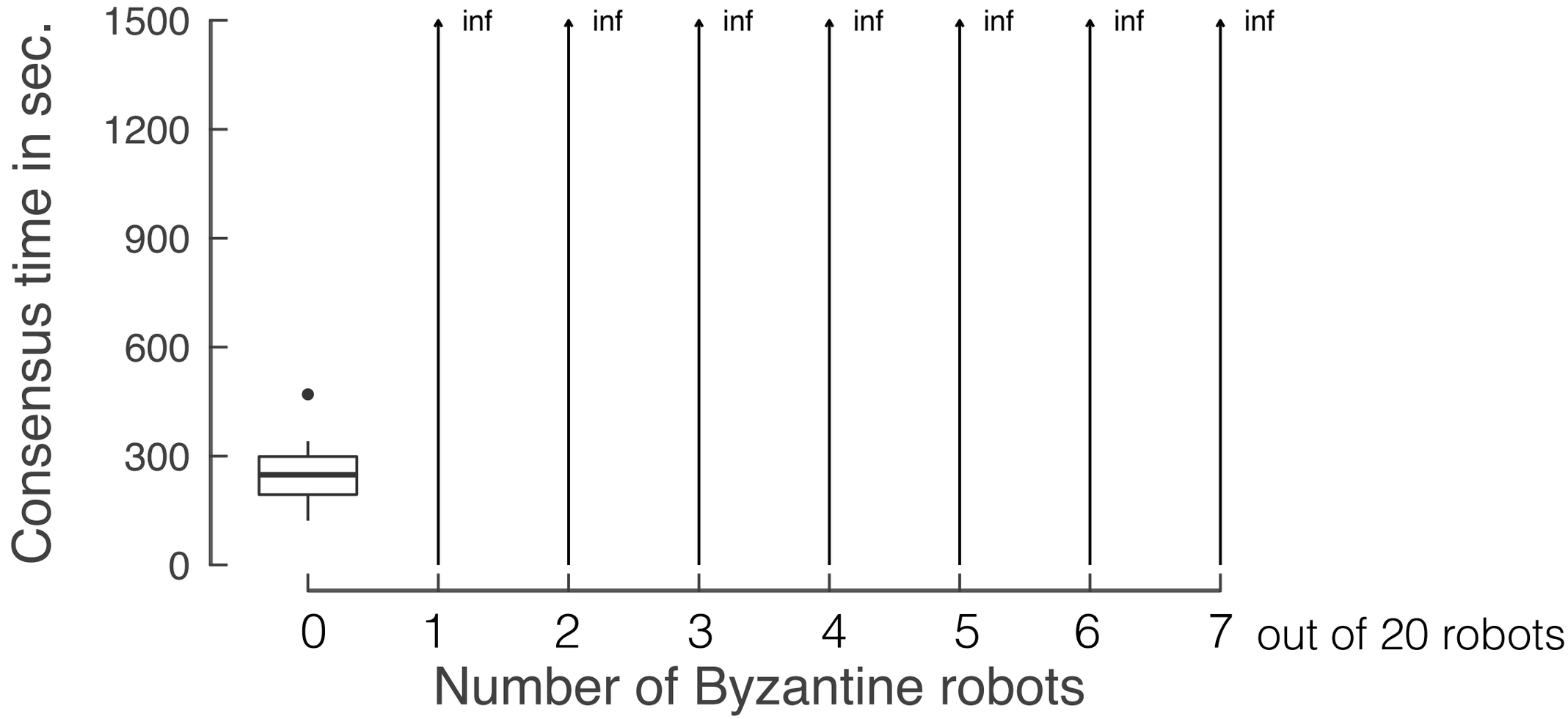
What if a robot keeps a constant estimate of 0% white tiles?

Byzantine

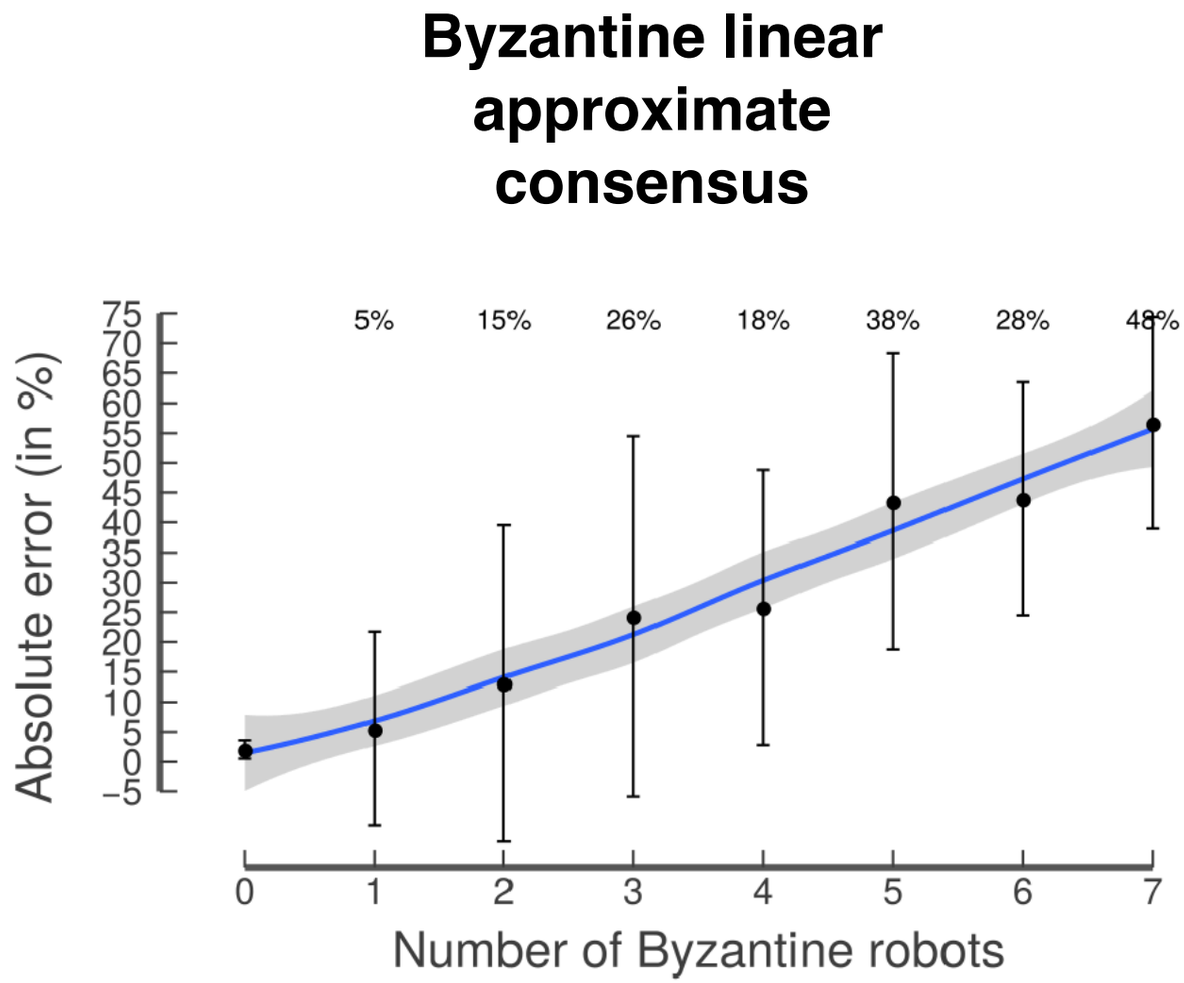


0% white tiles

Consensus time
(wait until all robots have approximately the same value)



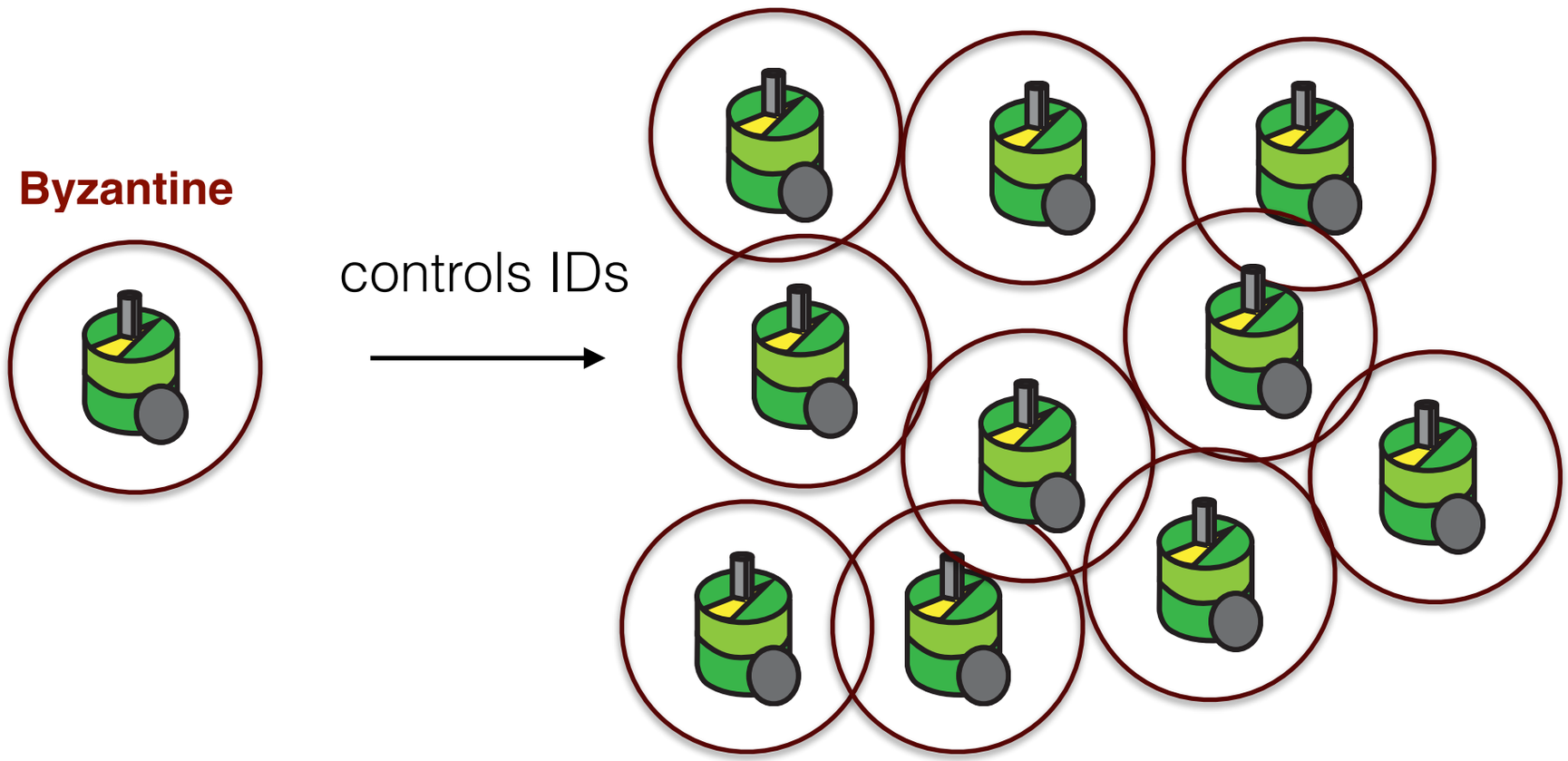
Let’s fix the runtime to 1,000 seconds



Randomly select one robot
(Byzantine or non-Byzantine)

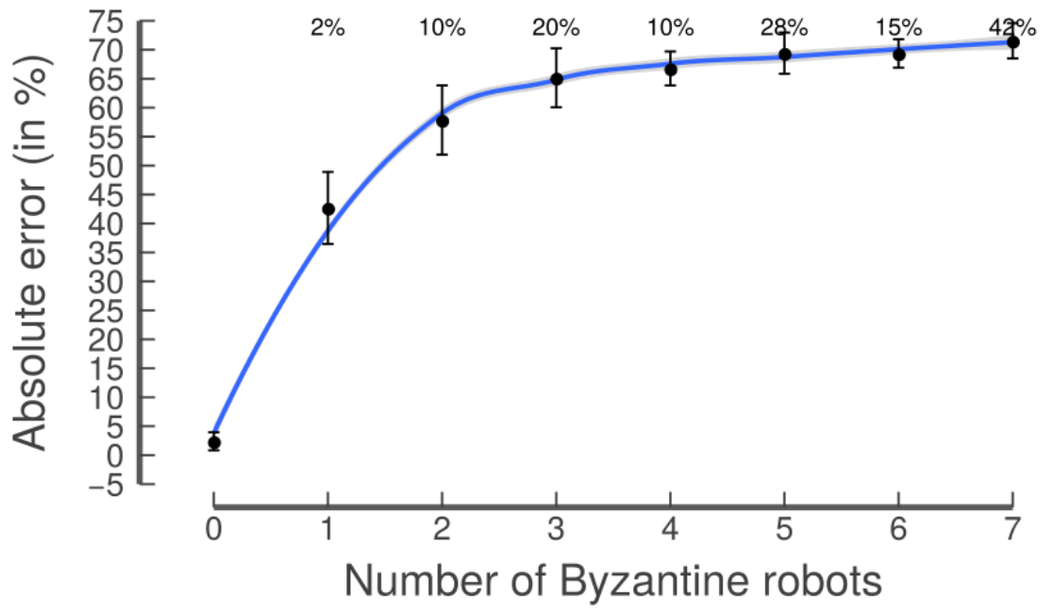
What if a robot performs a Sybil attack?

Robots can create new IDs on the fly
(Sybil attack)



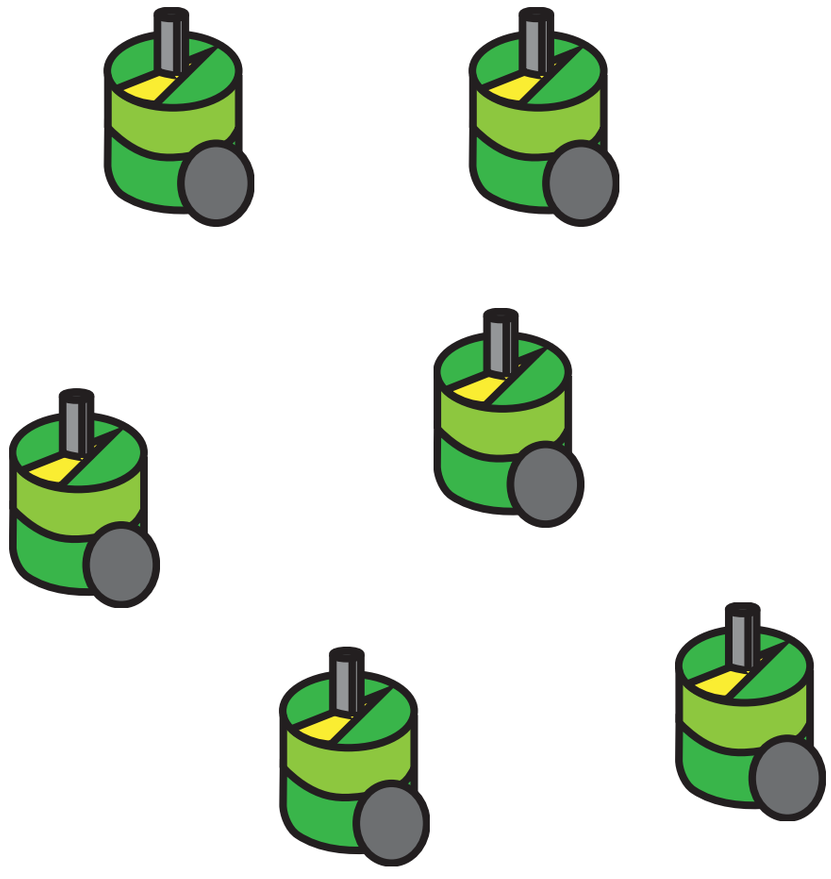
one robot can control many identities

Byzantine linear approximate consensus

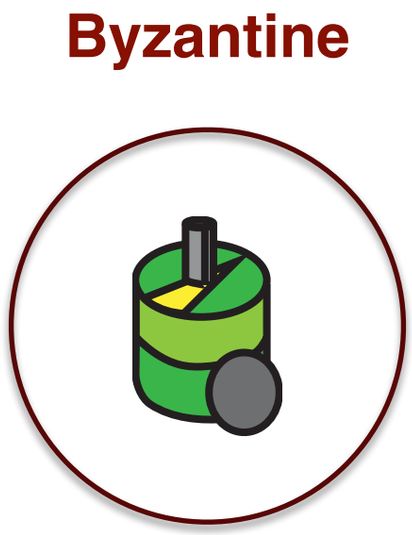


Randomly select one robot
(Byzantine or non-Byzantine)

To reach a consensus:
One Byzantine robot is stronger than the swarm



VS



0.0

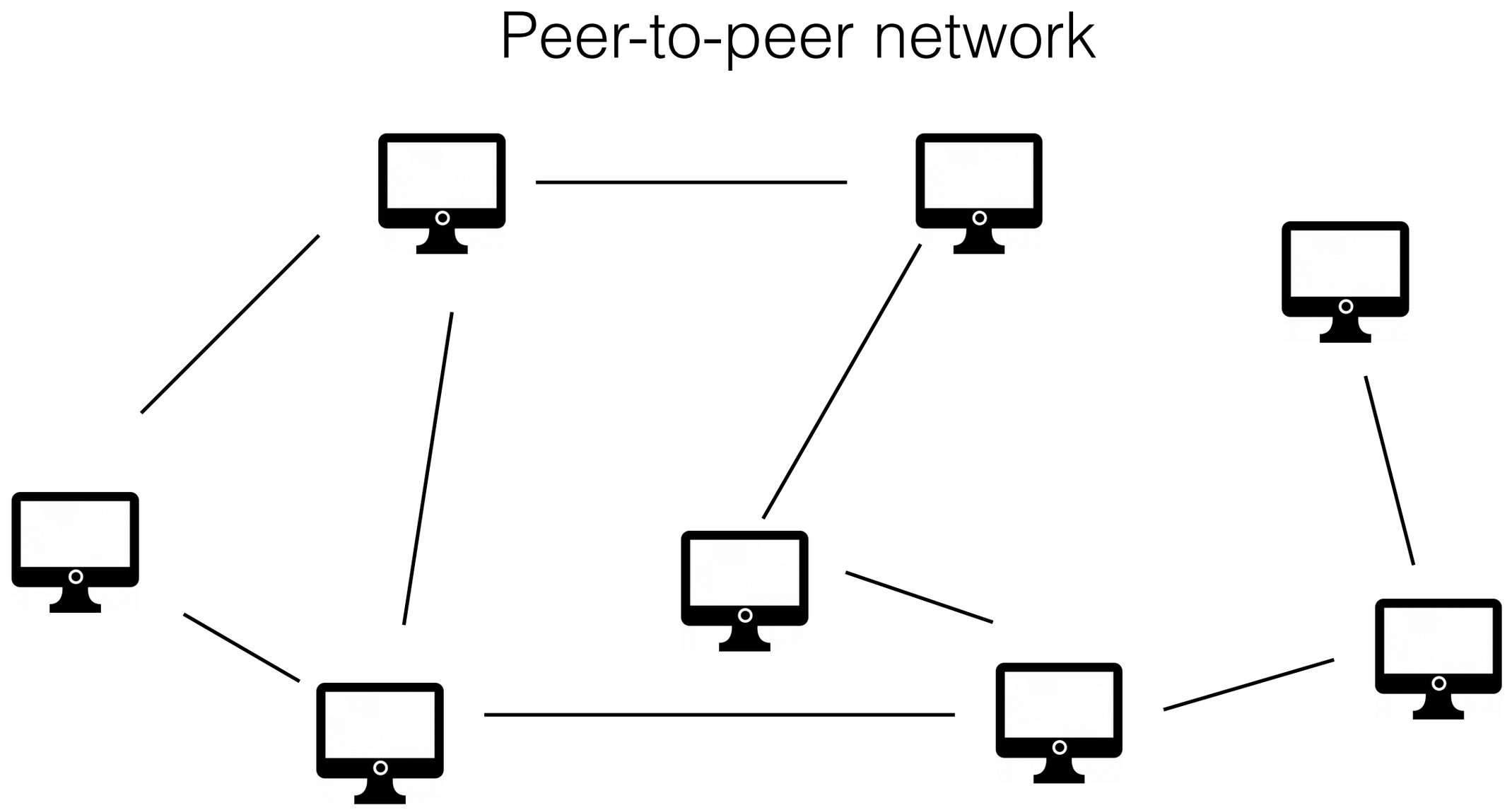
Security issues in swarm robotics

Part I
Shortcomings of classical approaches in swarm robotics

Part II
Blockchain technology

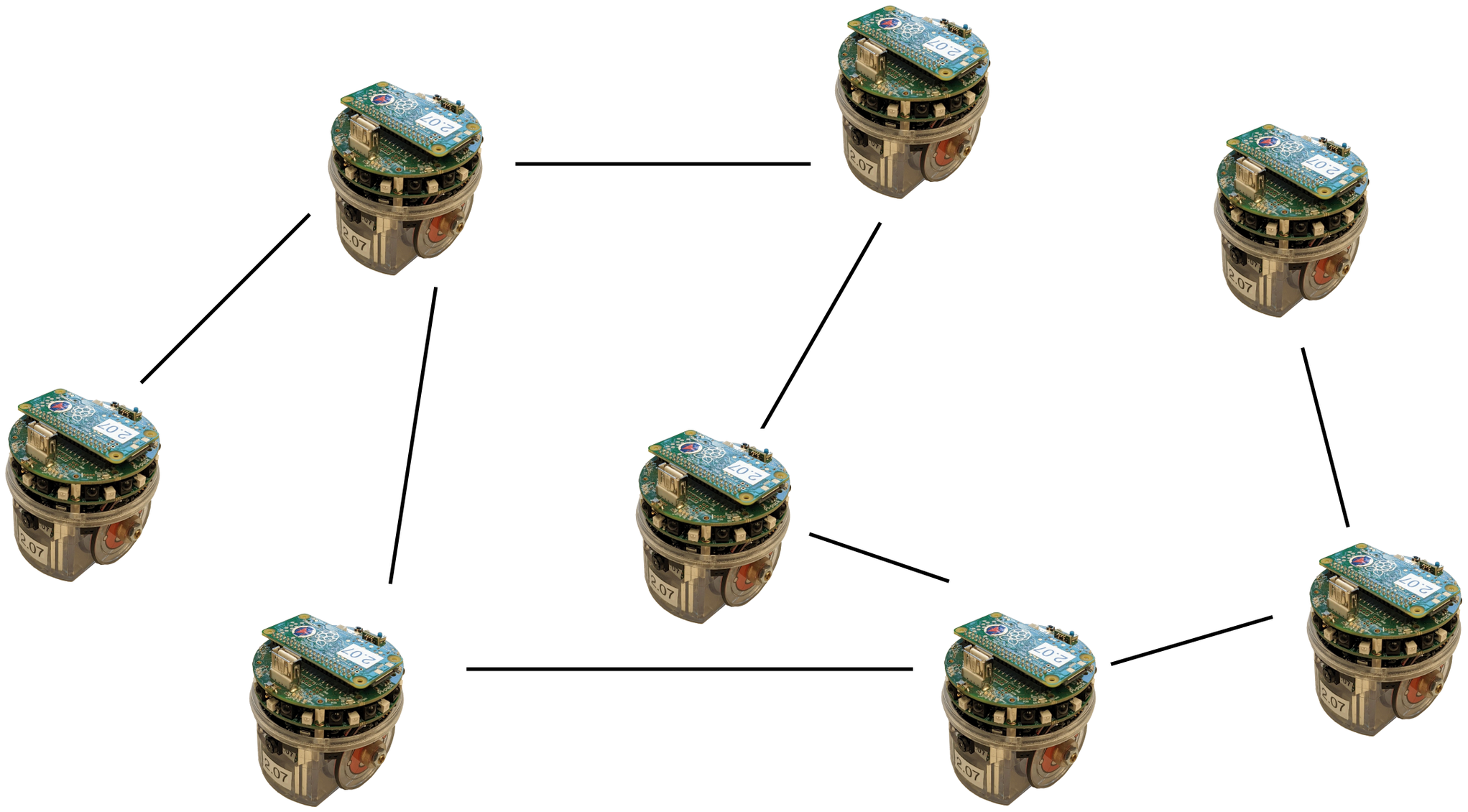
Part III
Blockchain-based smart contracts for the secure coordination of robot swarms

Blockchains make it possible to maintain trustworthy databases in peer-to-peer networks

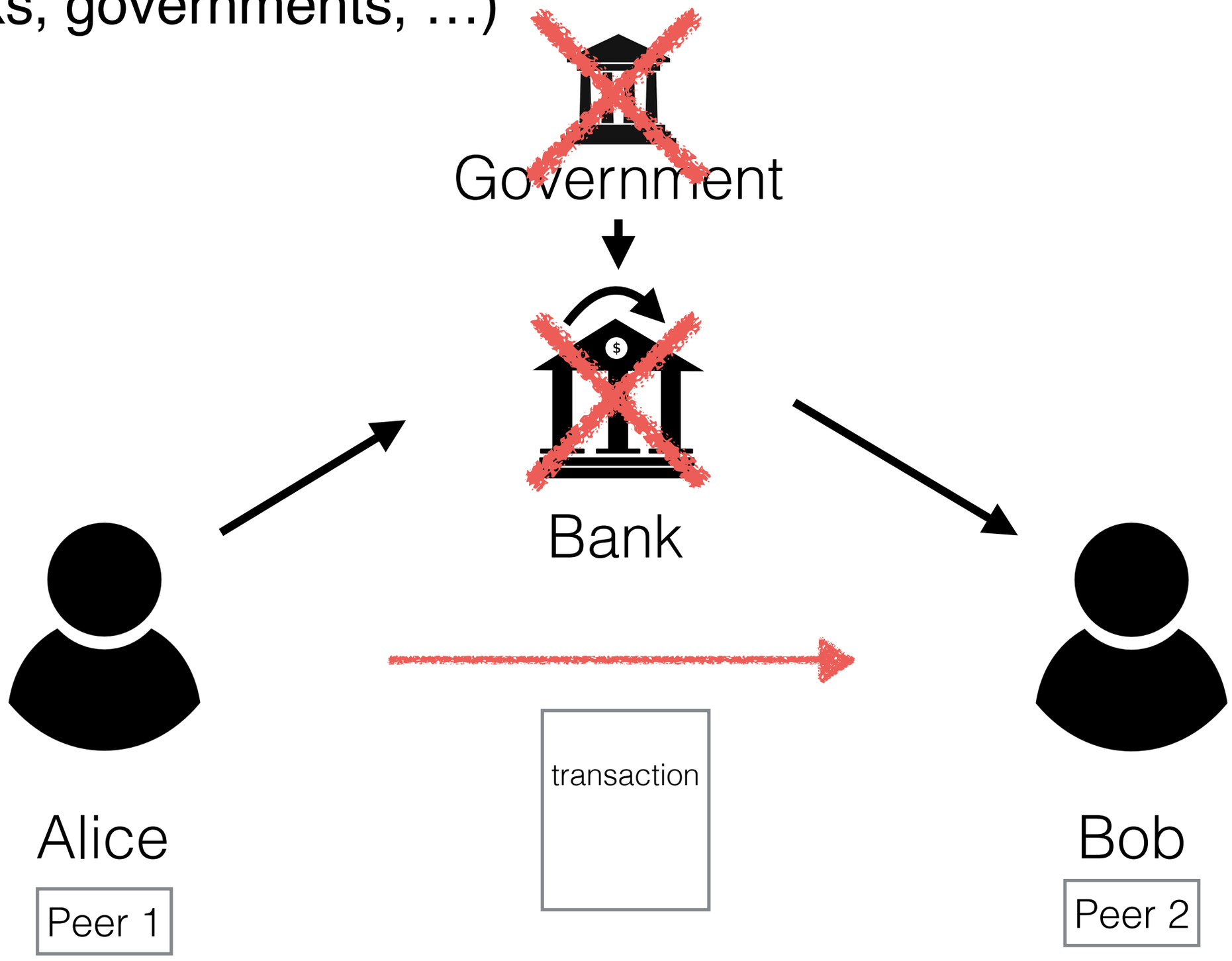


Blockchains make it possible to maintain trustworthy databases in peer-to-peer networks

Peer-to-peer network



Blockchain technology was originally developed to create digital currencies that do not require centralized control (banks, governments, …)



Why would you want to remove the bank?

save costs	transaction speed	don’t trust the bank
be anonymous	bank not reachable	ideological reasons
censorship	non-reversible transactions	micropayments
payments triggered by events	flexibility	…

Developing a peer-to-peer financial system poses many challenges:

- How to validate transactions?
- How to create new money?
- How to trust the system?
- How to prevent double-spending?
- How to reach consensus on who owns what?
- …?

Bitcoin: A Peer-to-Peer Electronic Cash System

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

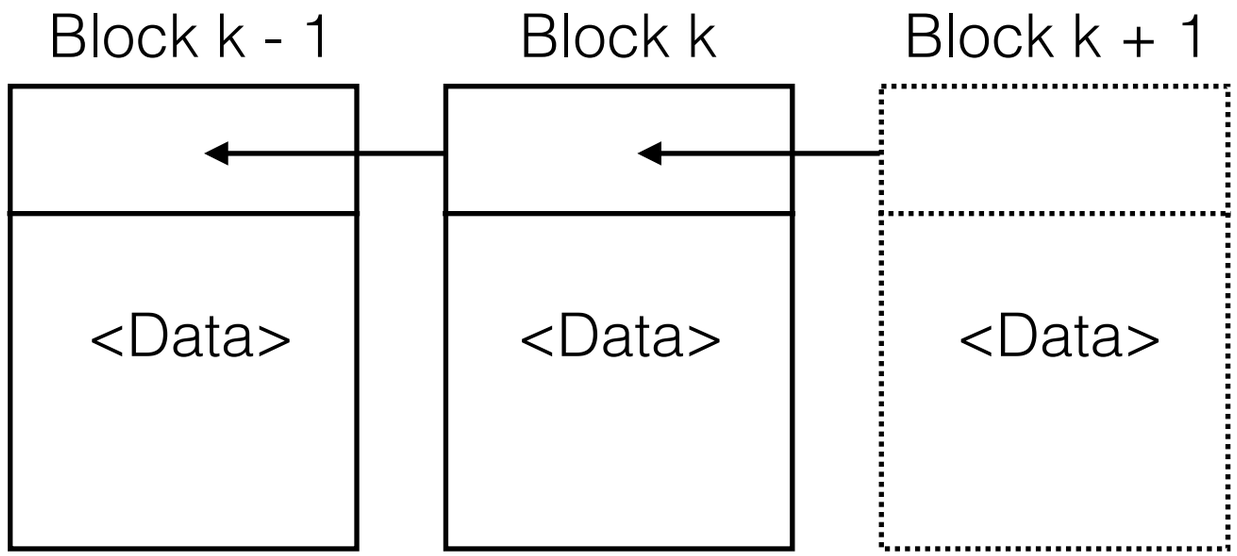
Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required 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.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the

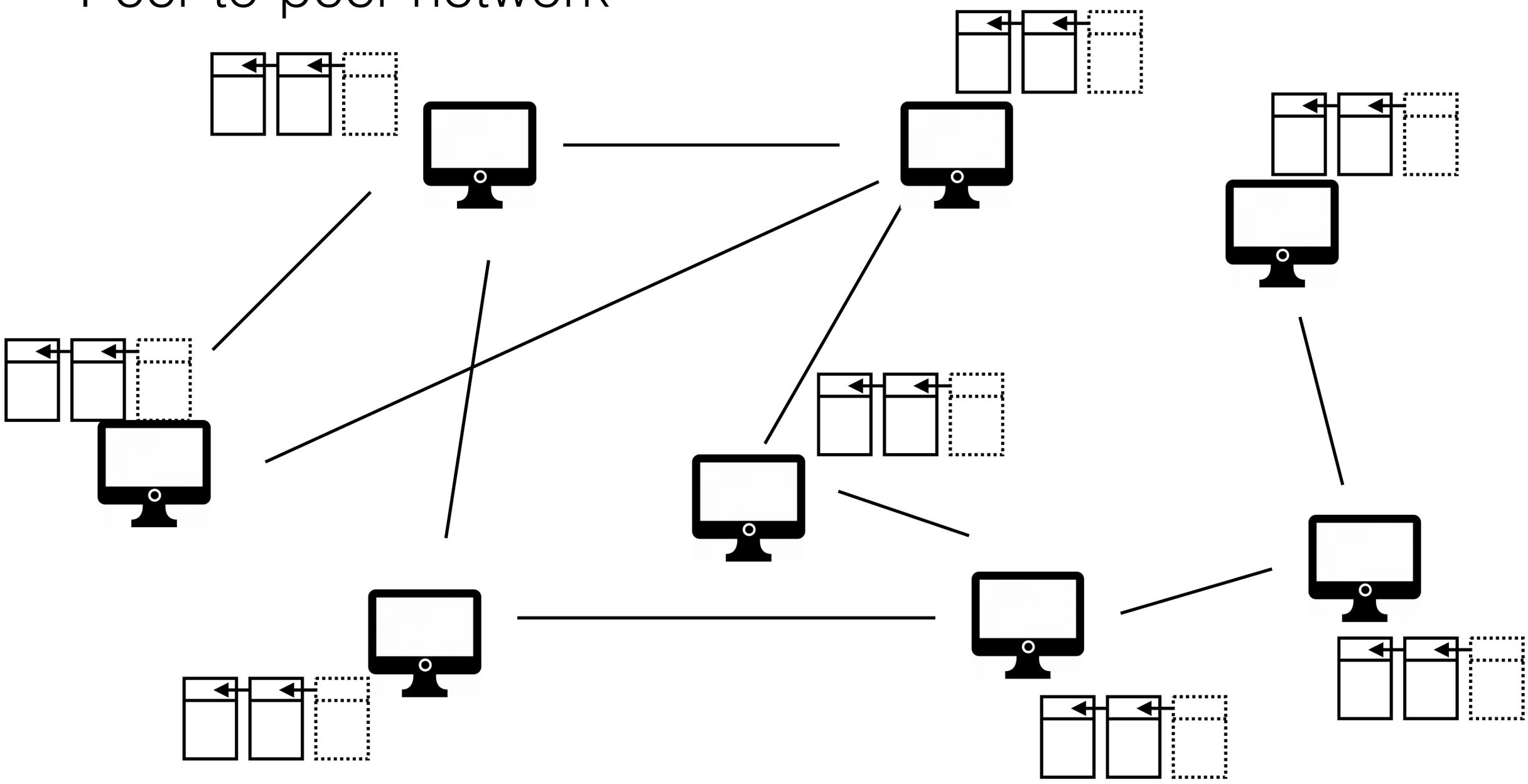

BANK

A blockchain is a database that consists of linked blocks



Every participant keeps keeps a local copy of the blockchain

Peer-to-peer network



Where does the <Data> come from?

transaction

Send 5 bitcoins (crypto tokens) to Bob

Alice

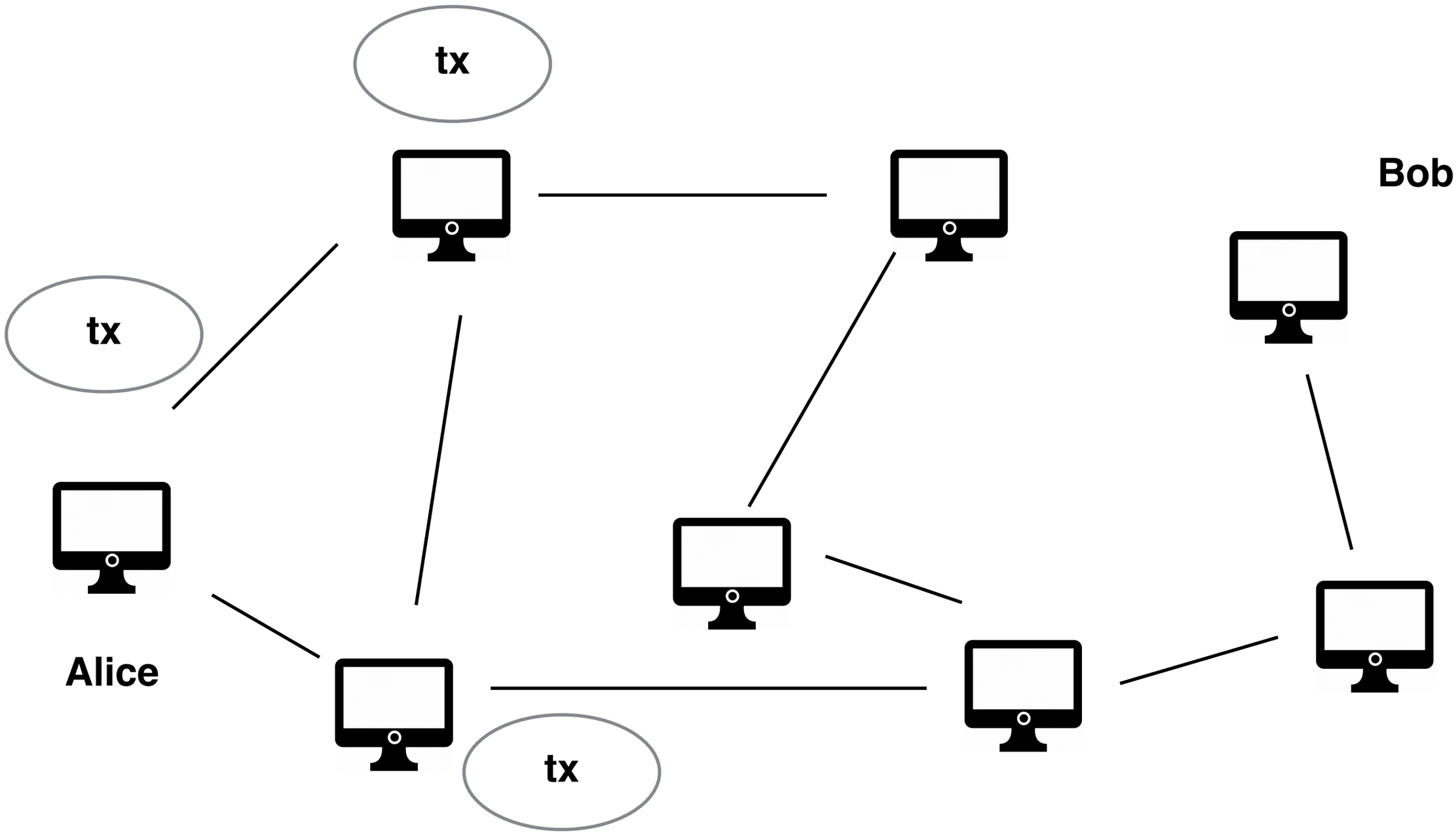
transaction

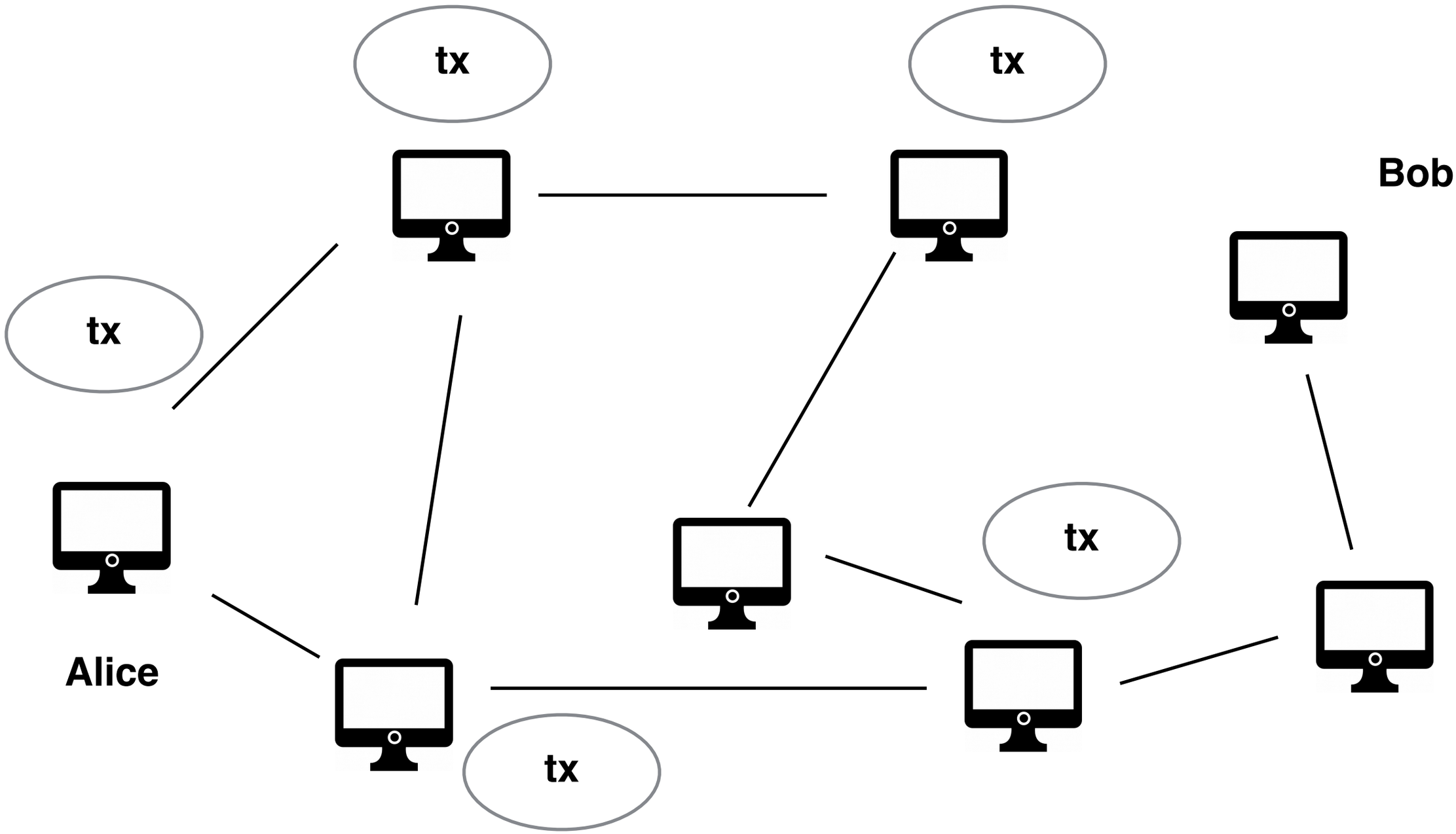
Send 5 bitcoins (crypto tokens) to User B

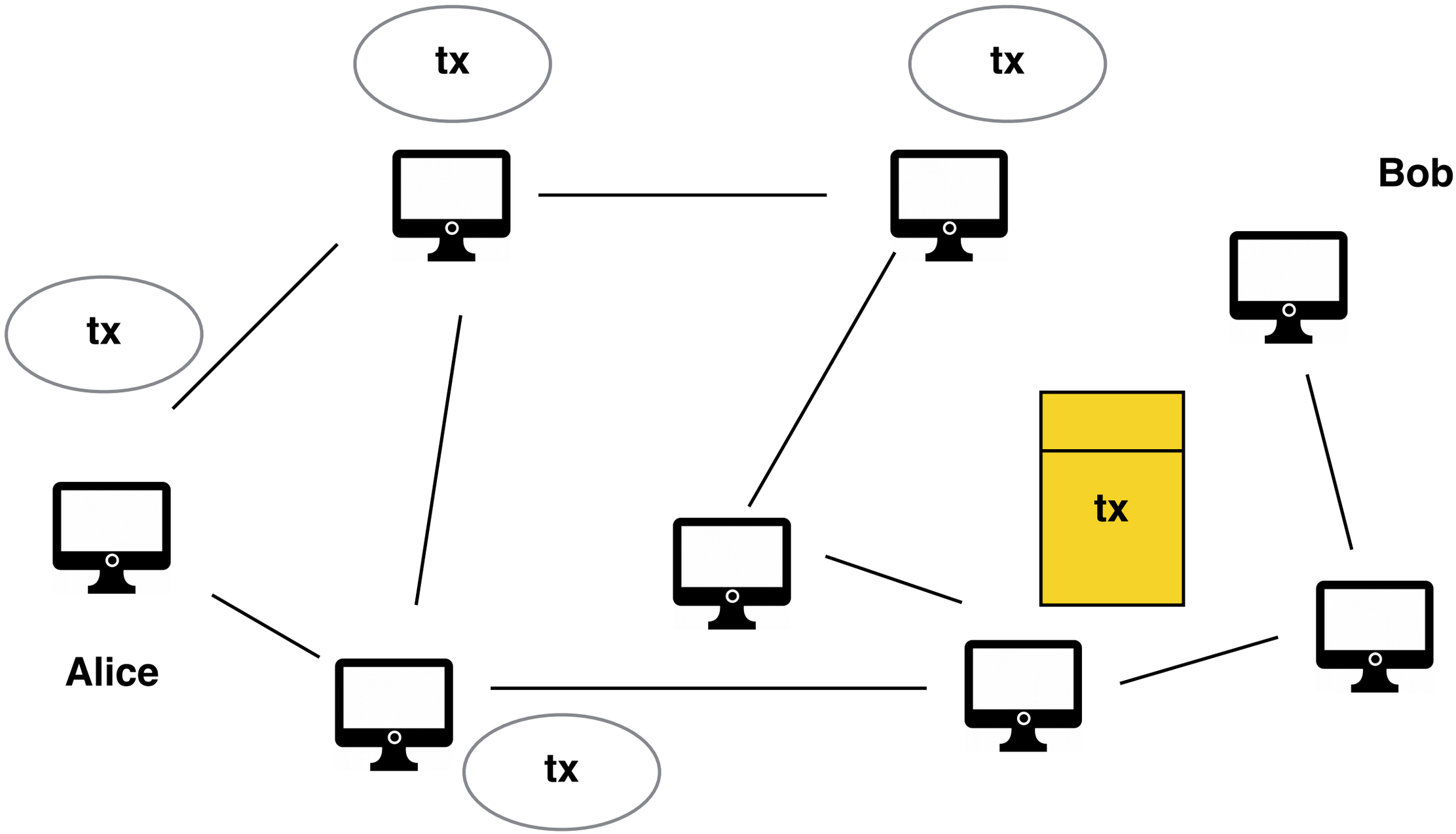
tx

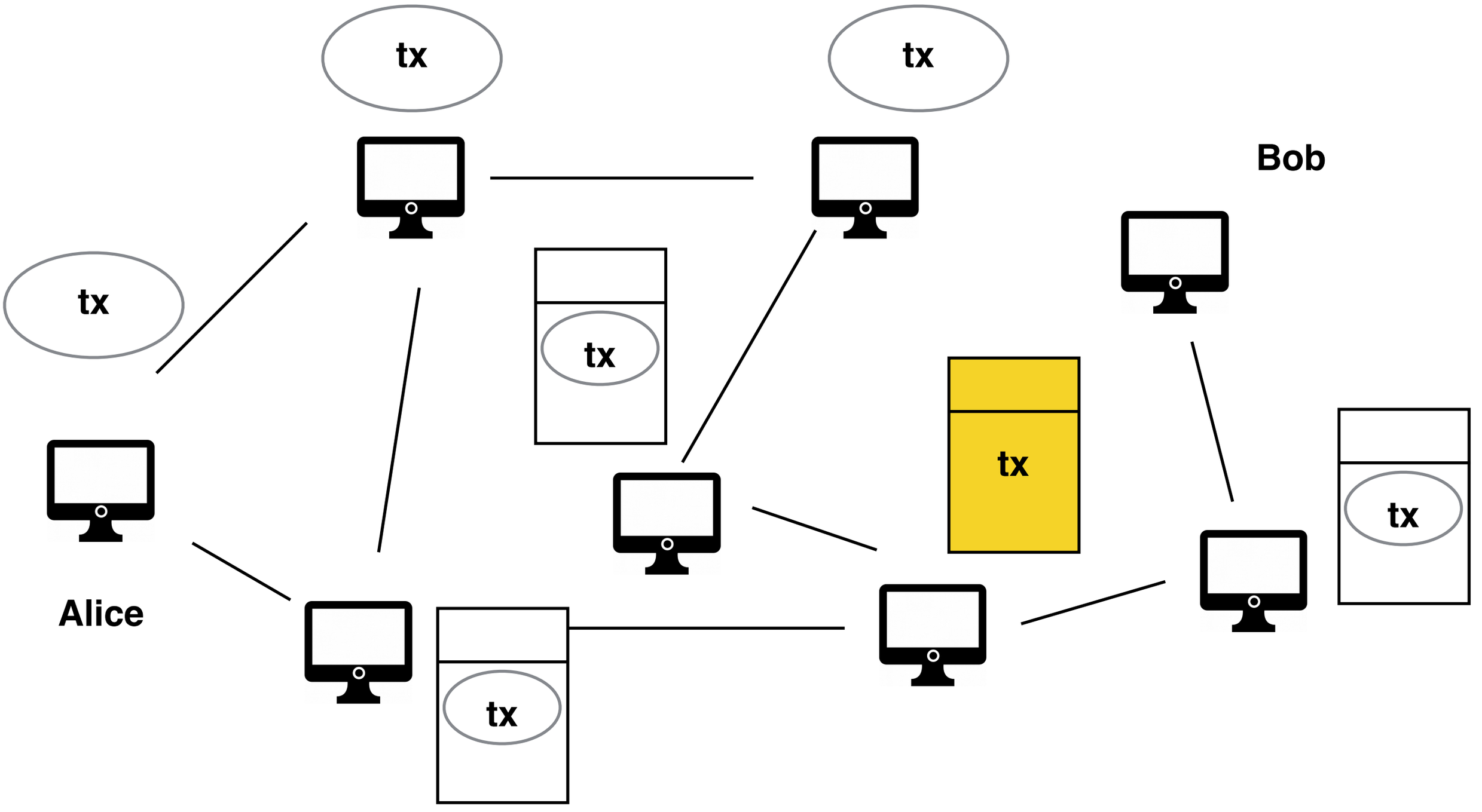
Alice

Bob


tx
Bob
tx
Alice
tx


tx
tx
Bob
tx
tx
Alice
tx


tx
tx
Bob
tx
tx
Alice
tx


tx
tx
Bob
tx
tx
tx
tx
tx
Alice

Send 5 bitcoins
(crypto tokens)
to Bob

tx

tx

tx

Bob

tx

tx

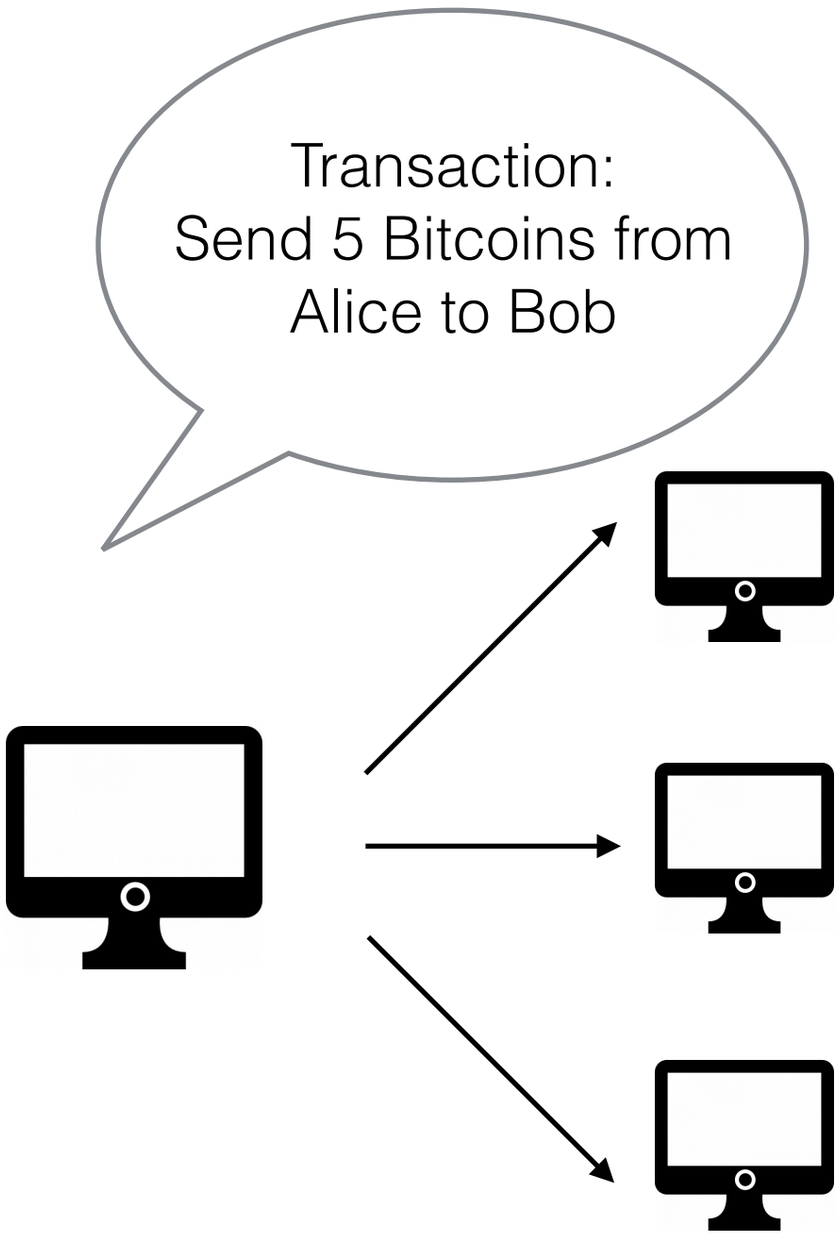
tx

tx

tx

Alice

To transfer Bitcoin tokens, participants publicly announce transactions



Is this transaction really from Alice?

-> Check signature

Digital Signatures

Public-key cryptography

document

signature

private/secret key (sk)

public key (pk)

sign(document , private key) -> signature

verify(document , public key, signature) -> boolean

A brief example of digital signatures

Generate private/public key pair:

```
gpg --full-generate-key
```

Distribute public key:

```
gpg --armor --output volker.gpg --export
```

Import public key (on another machine):

```
gpg --import volker.gpg
```

Create message:

```
echo "hello blockchain" > msg.txt
```

Sign message:

```
gpg --clearsign -o sig.gpg msg.txt
```

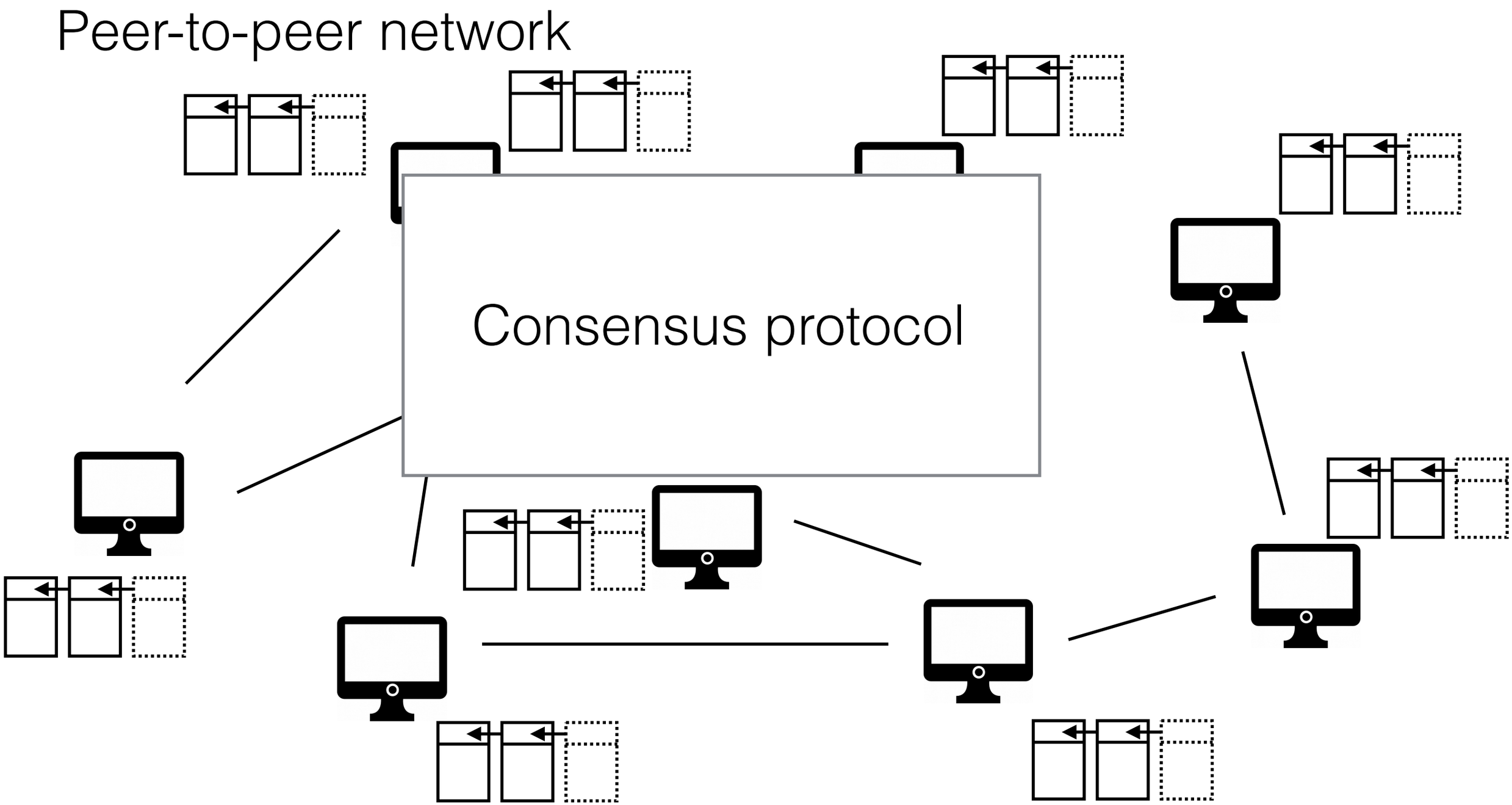
Send sig.gpg via email

Verify message (on another machine):

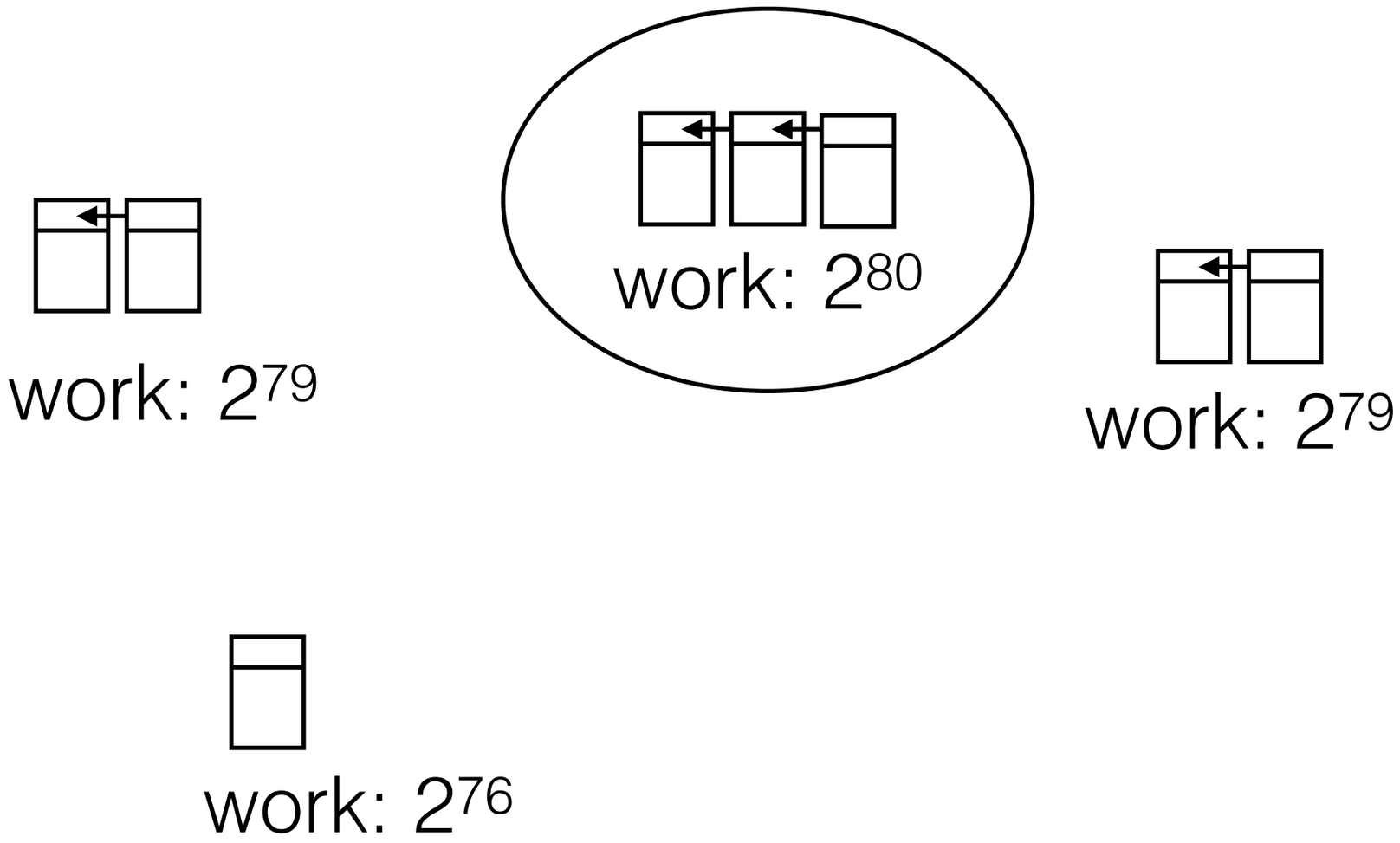
```
gpg --verify sig.gpg
```

How to ensure that everyone has the same version of the blockchain?

How to add new blocks which contain transactions?



The longest blockchain is accepted as the “true” blockchain



Work is the number of calculated hashes

Hash function: any function that can be used to map data (x) of arbitrary size to data of fixed size (y):
hash(x) = y

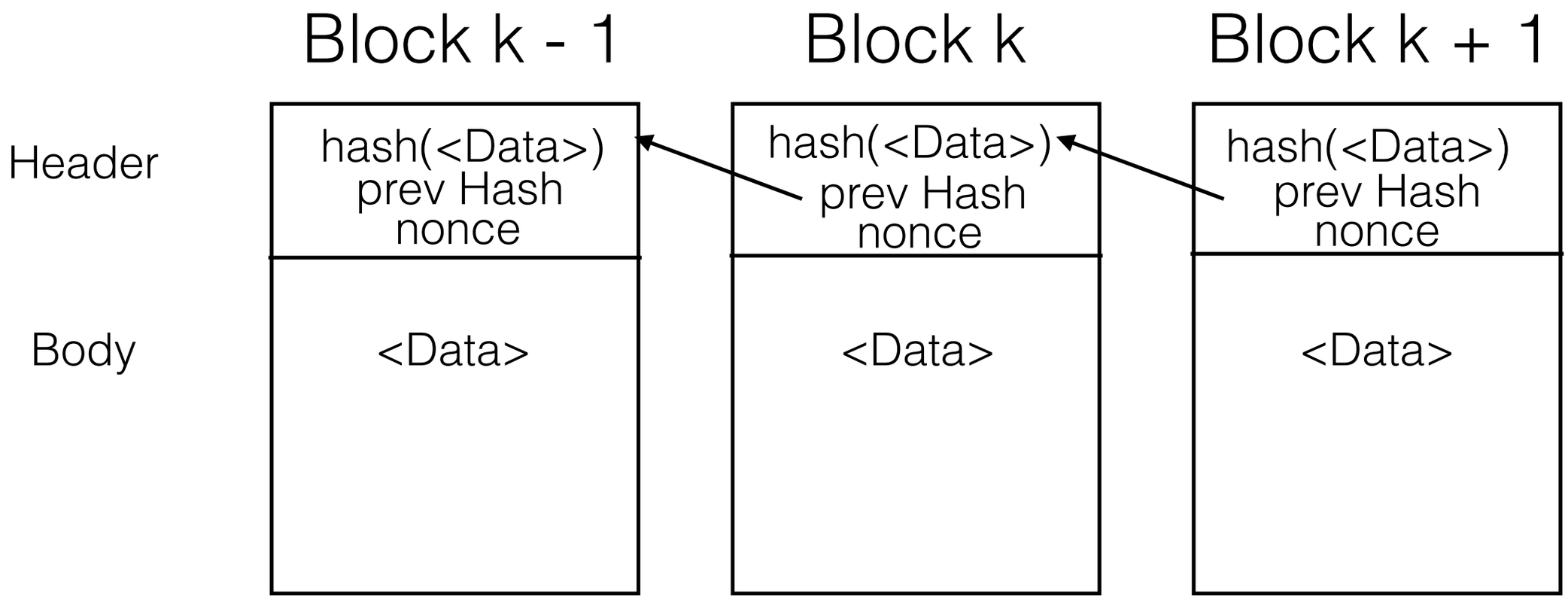
***Cryptographic* hash function**: “non-reversible” hash function:
“Given y, very hard to find x”

Mining (Proof of Work):

Find solution to a mathematical puzzle that can only be solved by brute force:

Find a suitable nonce value so that hashing the contents of a block plus that nonce leads a hash value starting with a pre-defined number of zeroes

The blockchain is build of blocks: chunks of data



Simple mining example in Python

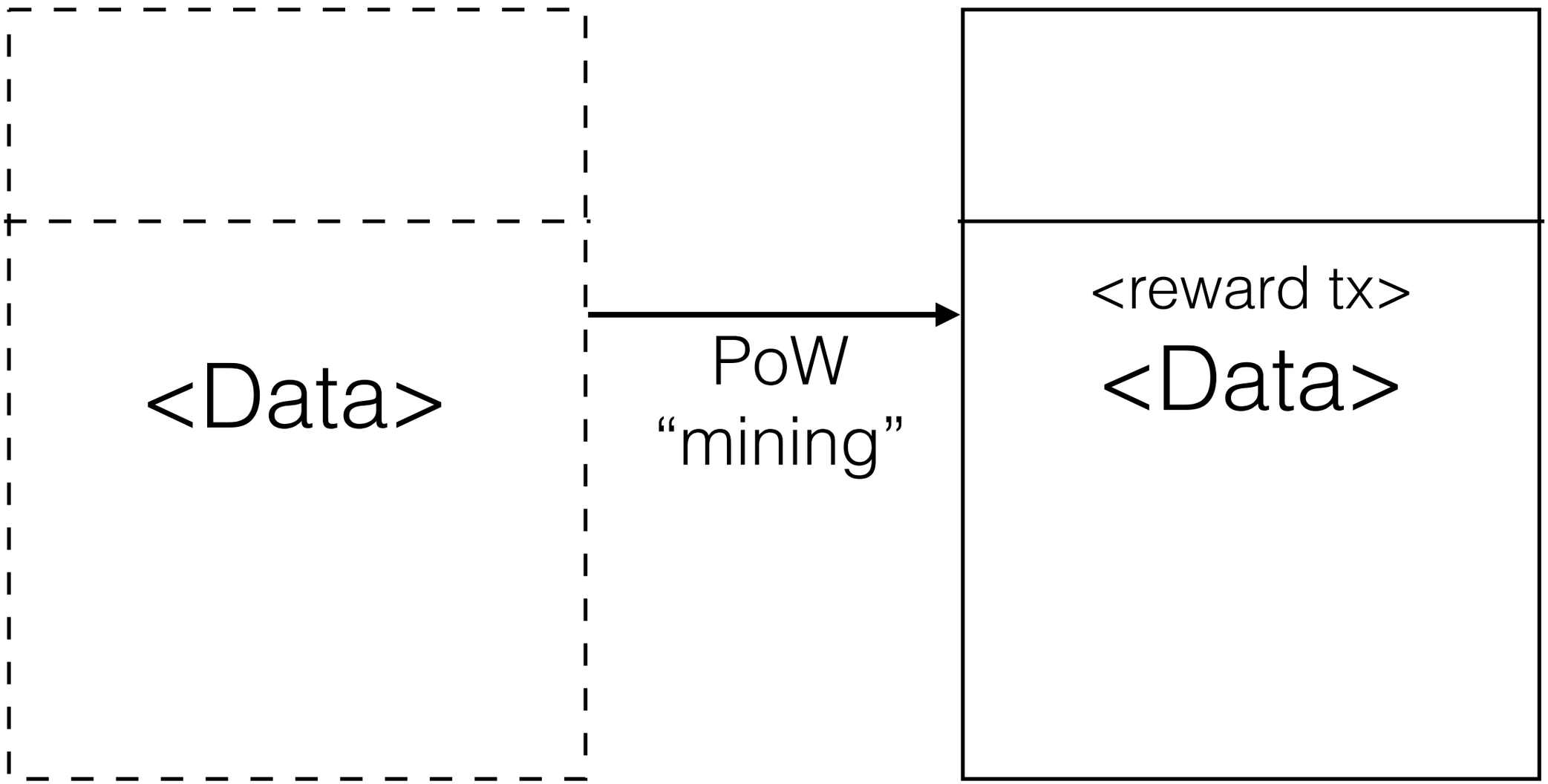
```python
import hashlib

difficulty = "00"

print("Difficulty is set to " + difficulty)

def my_sha256(transaction):

    print("Hashing: " + transaction)
    hash_result = hashlib.sha256(transaction.encode()).hexdigest()

    return hash_result


def mine(transaction):

    nonce = 0
    while True:

        hash_result = my_sha256(transaction + " " + str(nonce))

        if hash_result.startswith(difficulty):
            print("Found solution using nonce " + str(nonce))
            break
        nonce += 1
```

Mining (Proof of Work): Find solution to a mathematical puzzle that can only be solved by brute force:
hashing using different nonce values

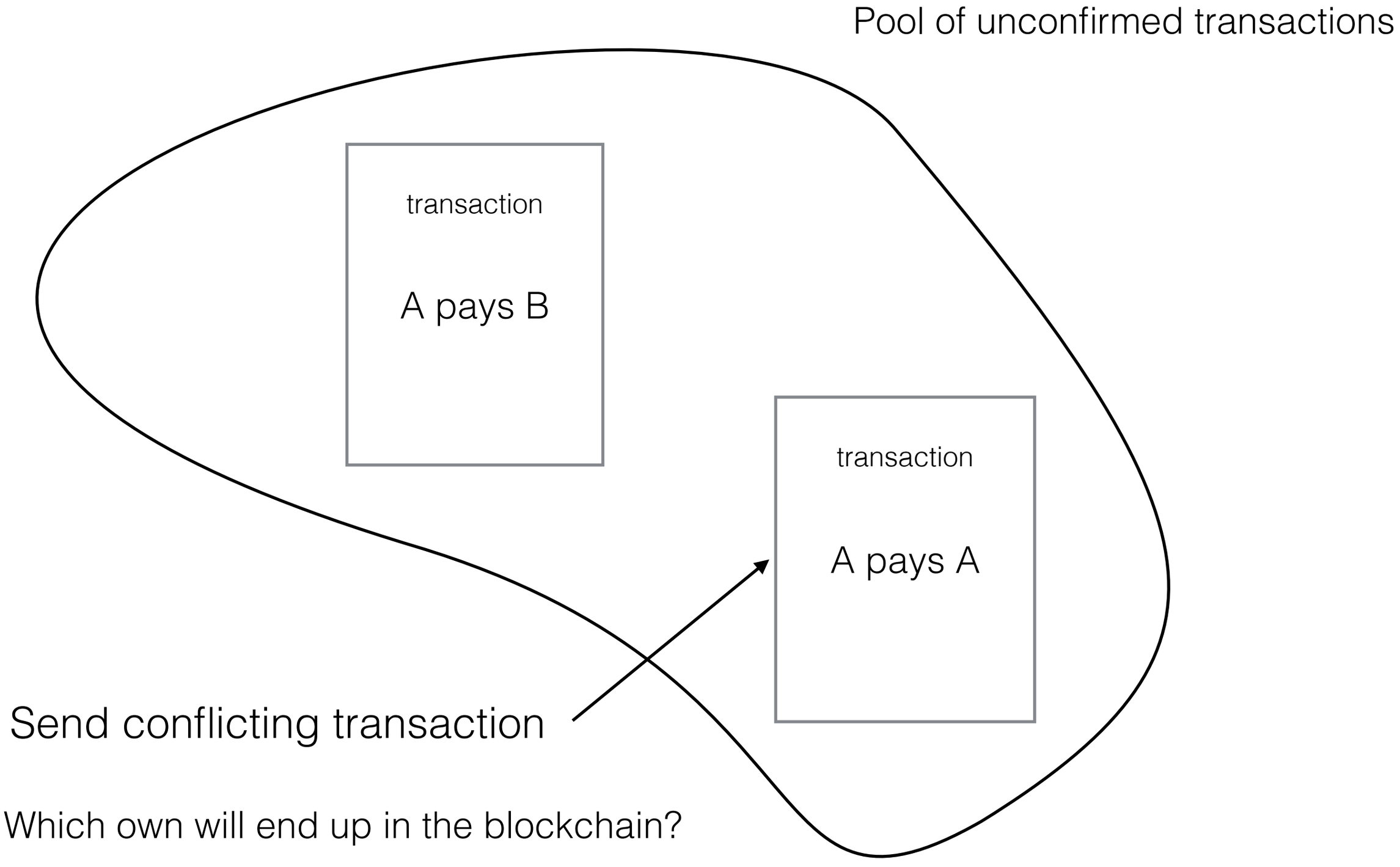
1. Consensus based on amount of mining work (longest chain)

2. Immutability: Changing content of blocks would require remining

Mining is a *race*:
As a miner you want to be the first to find the solution to get a reward

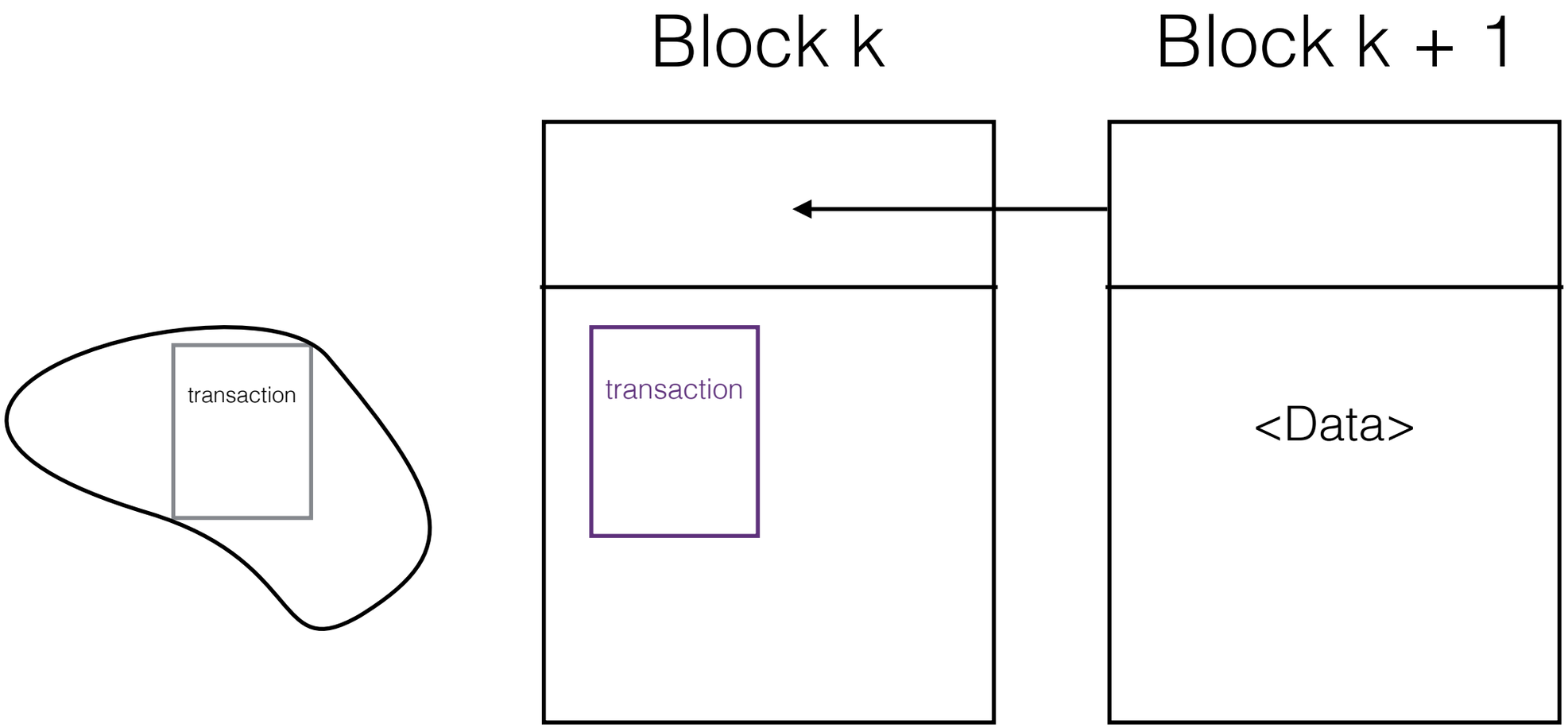


How to cheat/attack?

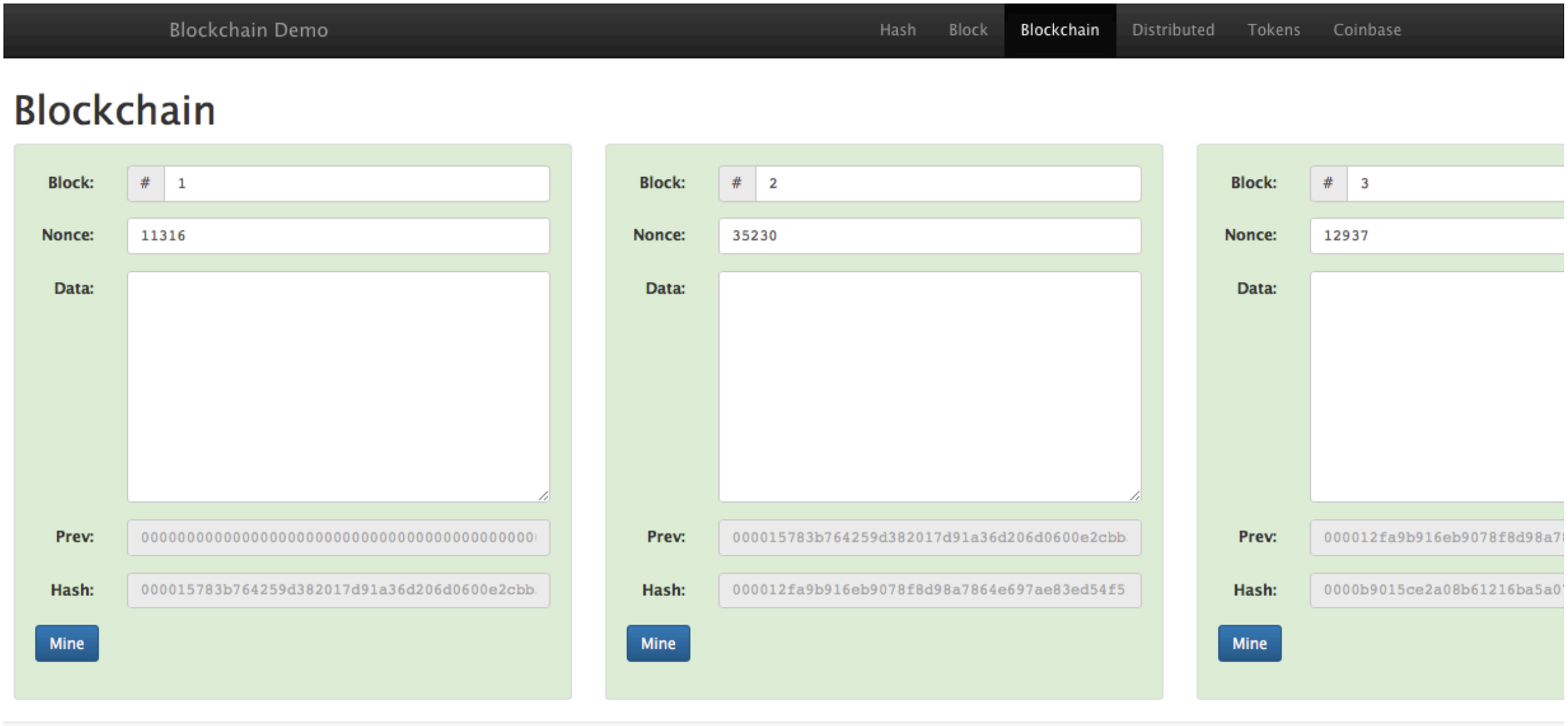


Which own will end up in the blockchain?

Wait for confirmations



Changing data in a previous block would require an attacker to redo the Proof-of-Work of all later blocks



https://andersbrownworth.com/blockchain/blockchain

https://andersbrownworth.com/blockchain/blockchain

Short summary:
The main building blocks of Bitcoin are:

- Peer-to-peer network

- Public database (blockchain)

- Digital Signatures

- Proof-of-Work (Mining)

Quite early, people realized that Bitcoin is just one application build on blockchain technology

Appearance of “Altcoins”

Image source: https://steemit.com/altcoins/@frieswiththat/top-10-sleeper-altcoins

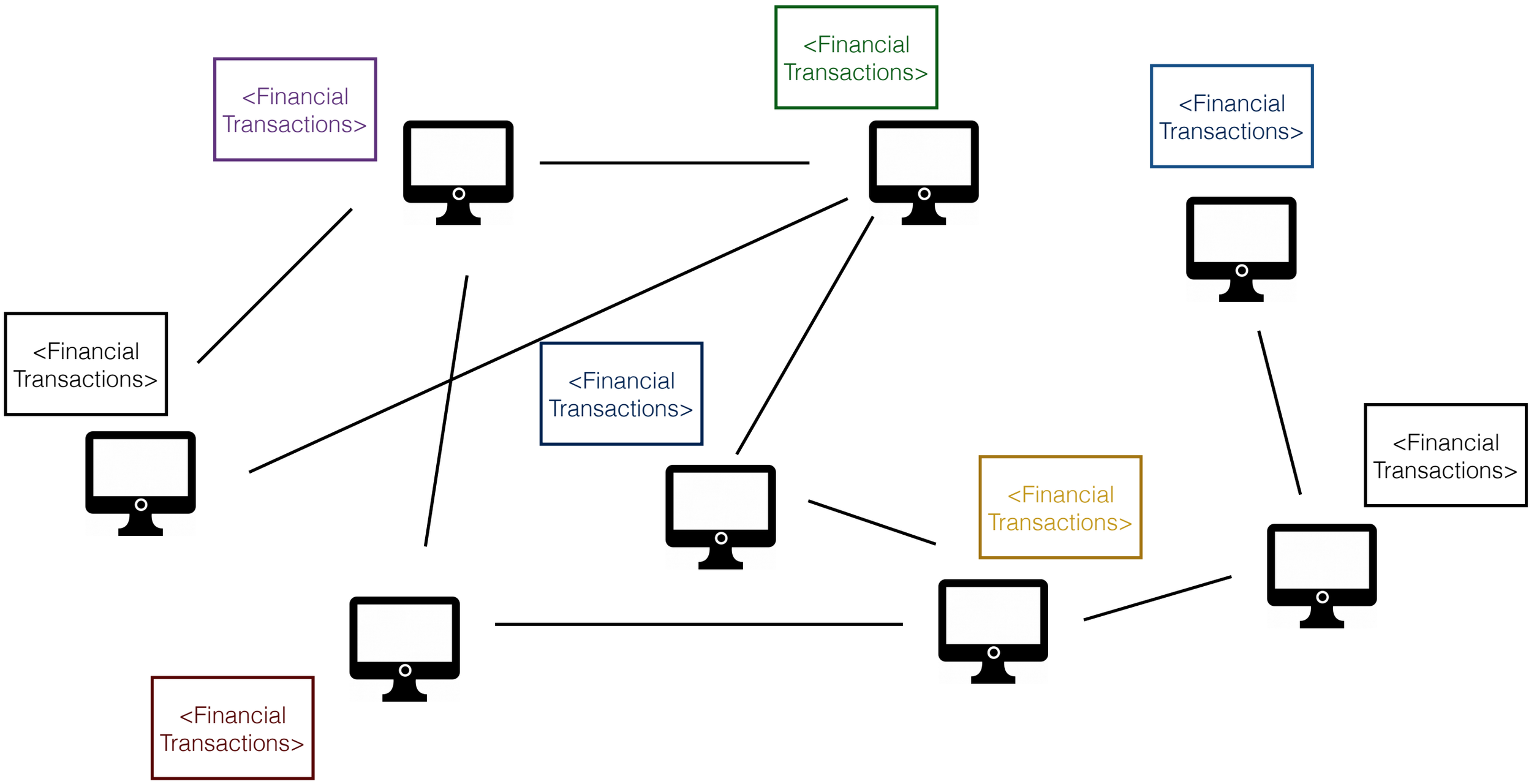
See also https://coinmarketcap.com/

Use cases of Altcoins

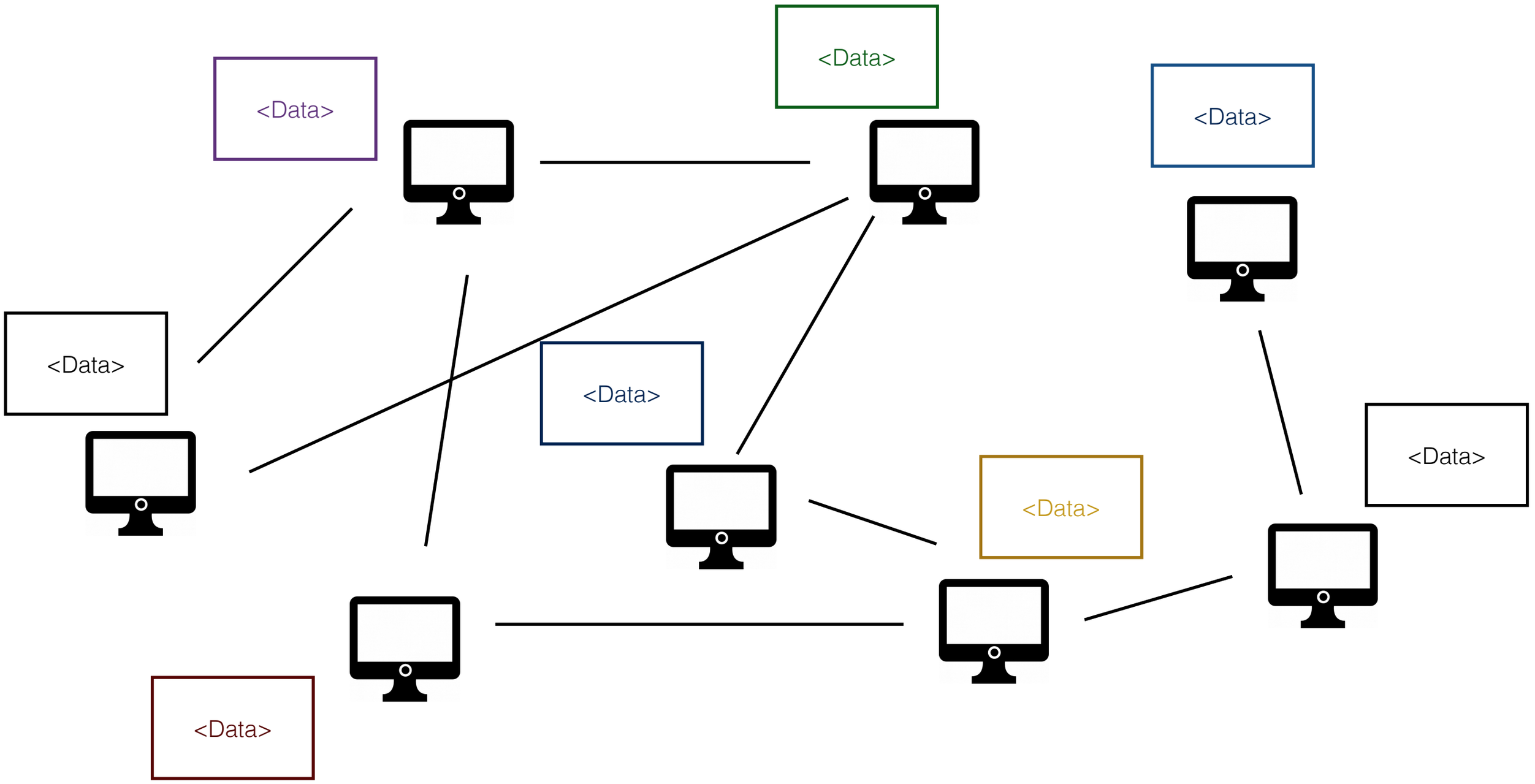
- financial records
- URL registration
- voting
- ...

Image source: https://steemit.com/news/@jesusvmorales/ethereum-eth-the-best-currency-to-buy-and-mine

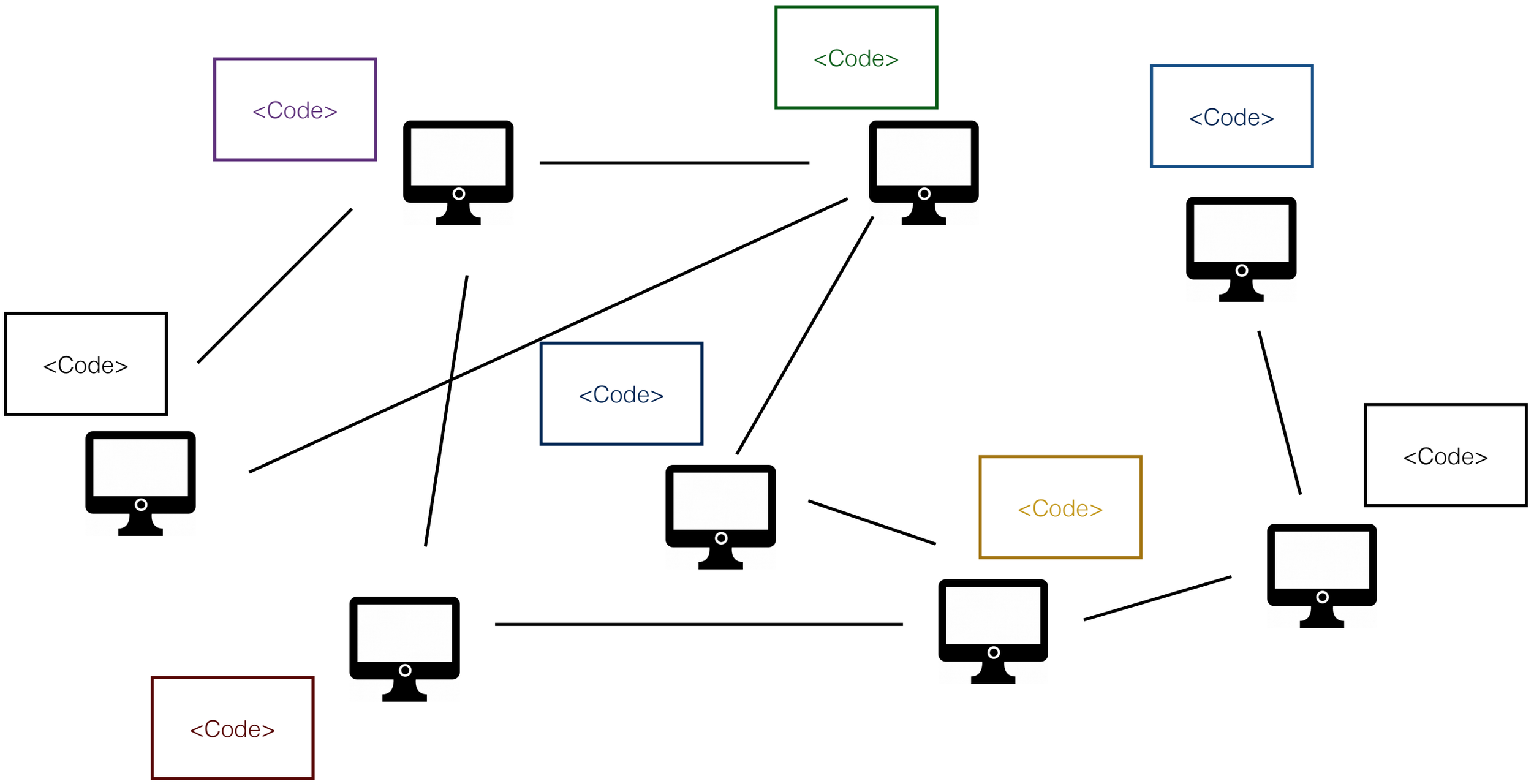
Peer-to-peer network



Peer-to-peer network



Peer-to-peer network



Blockchain 1.0
(2009)

Blockchain 2.0
(2015)

Blockchain-based smart contracts

Programming code stored on a blockchain and executed by each participant

ethereum

Uncompiled smart contract

```
pragma solidity ^0.4.0;
contract Estimation {

int public mean;
int public count;
int public threshold = 100000;
int public m2;
int public pub_se;
int W_n;

mapping(address => int) public weights;

event consensusReached(uint c);


 function sqrt(int x) constant  returns (int y) {
   int z = (x + 1) / 2;
   y = x;
   while (z < y) {
     y = z;
     z = (x / z + z) / 2;
   }
 }

function abs(int x) returns (int y) {
    if (x < 0) {
        return -x;
```

Uncompiled smart contract

```
pragma solidity ^0.4.0;
contract Estimation {

int public mean;
int public count;
int public threshold = 100000;
int public m2;
int public pub_se;
int W_n;

mapping(address => int) public weights;

event consensusReached(uint c);


 function sqrt(int x) constant  returns (int y) {
   int z = (x + 1) / 2;
   y = x;
   while (z < y) {
     y = z;
     z = (x / z + z) / 2;
   }
 }

function abs(int x) returns (int y) {
    if (x < 0) {
        return -x;
```

compile

```
PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x18
PUSH1 0x0
SSTORE
CALLVALUE
ISZERO
PUSH1 0x13
JUMPI
PUSH1 0x0
DUP1
```

Compiled smart contract

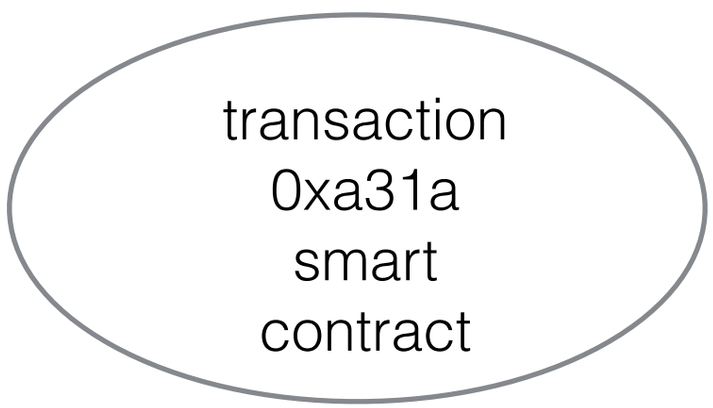
Compiled smart contract

```
PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x18
PUSH1 0x0
SSTORE
CALLVALUE
ISZERO
PUSH1 0x13
JUMPI
PUSH1 0x0
DUP1
```

Compiled
smart contract

PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x18
PUSH1 0x0
SSTORE
CALLVALUE
ISZERO
PUSH1 0x13
JUMPI
PUSH1 0x0
DUP1

publish



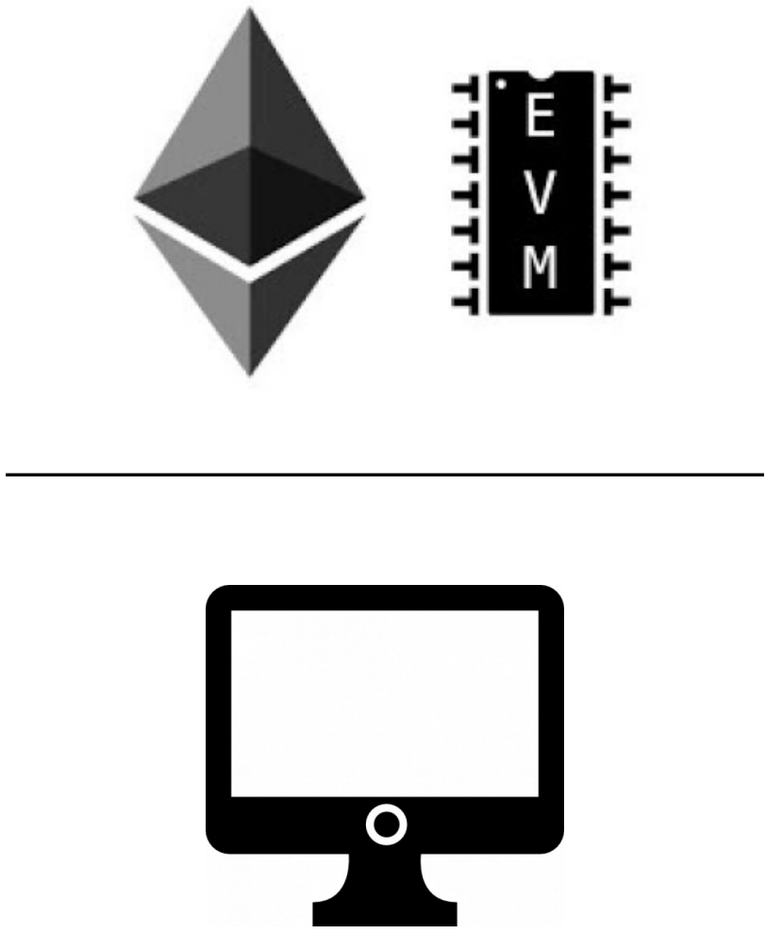
Compiled
smart contract

Block k

PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x18
PUSH1 0x0
SSTORE
CALLVALUE
ISZERO
PUSH1 0x13
JUMPI
PUSH1 0x0
DUP1

Block k

PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x18
PUSH1 0x0
SSTORE
CALLVALUE
ISZERO
PUSH1 0x13
JUMPI
PUSH1 0x0
DUP1



Financial transactions

Bitcoin

Blockchain 1.0

Image source: https://bitcoin.org/de/

Arbitrary decentralized applications

Ethereum

Blockchain 2.0

Use case of Ethereum:

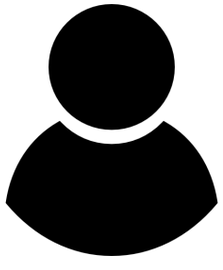
ethereum

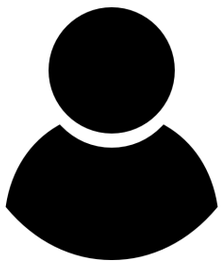
Blockchain-based smart contracts:
programming code executed via blockchain technology

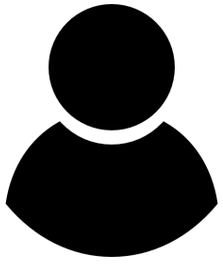
→

tamper-proof code
“without any possibility of downtime, censorship, fraud or third-party interference” (ethereum.org)

“Traditional” crowdfunding







backers

centralized crowdfunding platform (e.g. kickstarter.com)

needs **trust**

all-or-nothing

project (e.g. smartwatch)

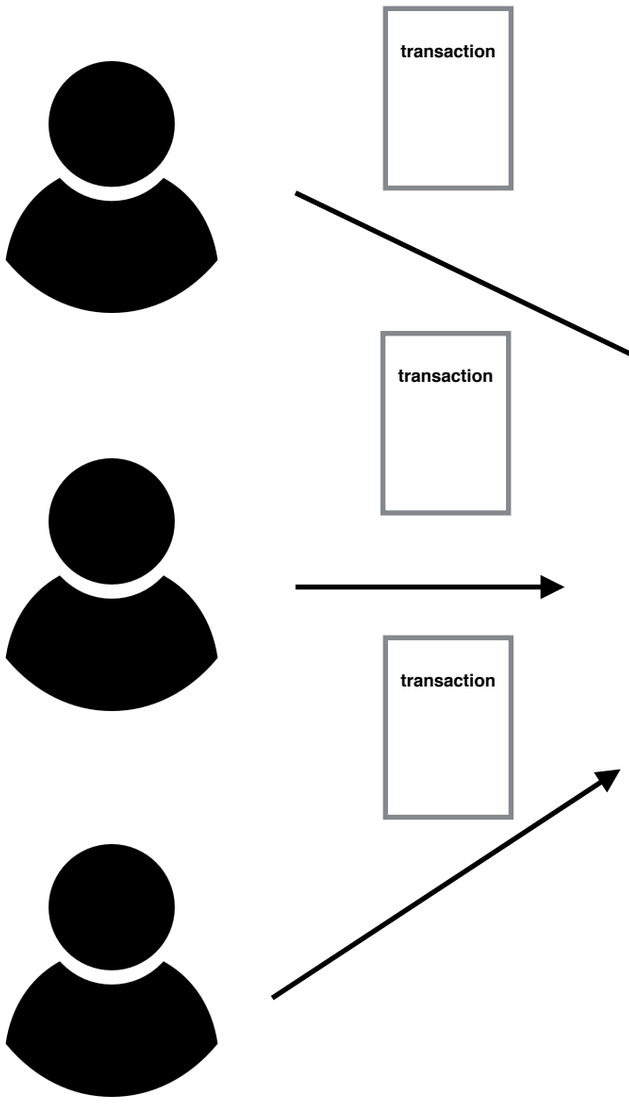
needs **trust**

- money really sent to the project?
- transparency
- potentially high fees

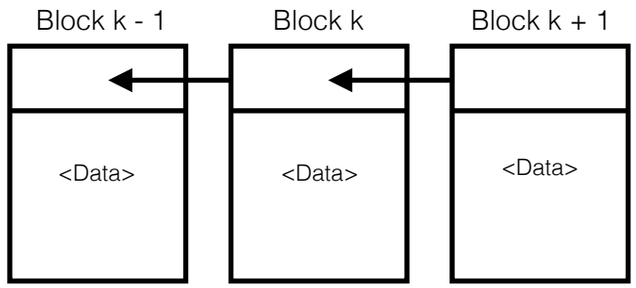
Image source:https://www.visa.ca/en_CA/pay-with-visa/cards/credit-cards.html

Ethereum-based crowdfunding

(see https://ethereum.org/crowdsale for an implementation in Solidity)



backers



decentralized crowdfunding platform based on Ethereum

all-or-nothing

project (e.g. smartwatch)

needs **trust**

trustless:

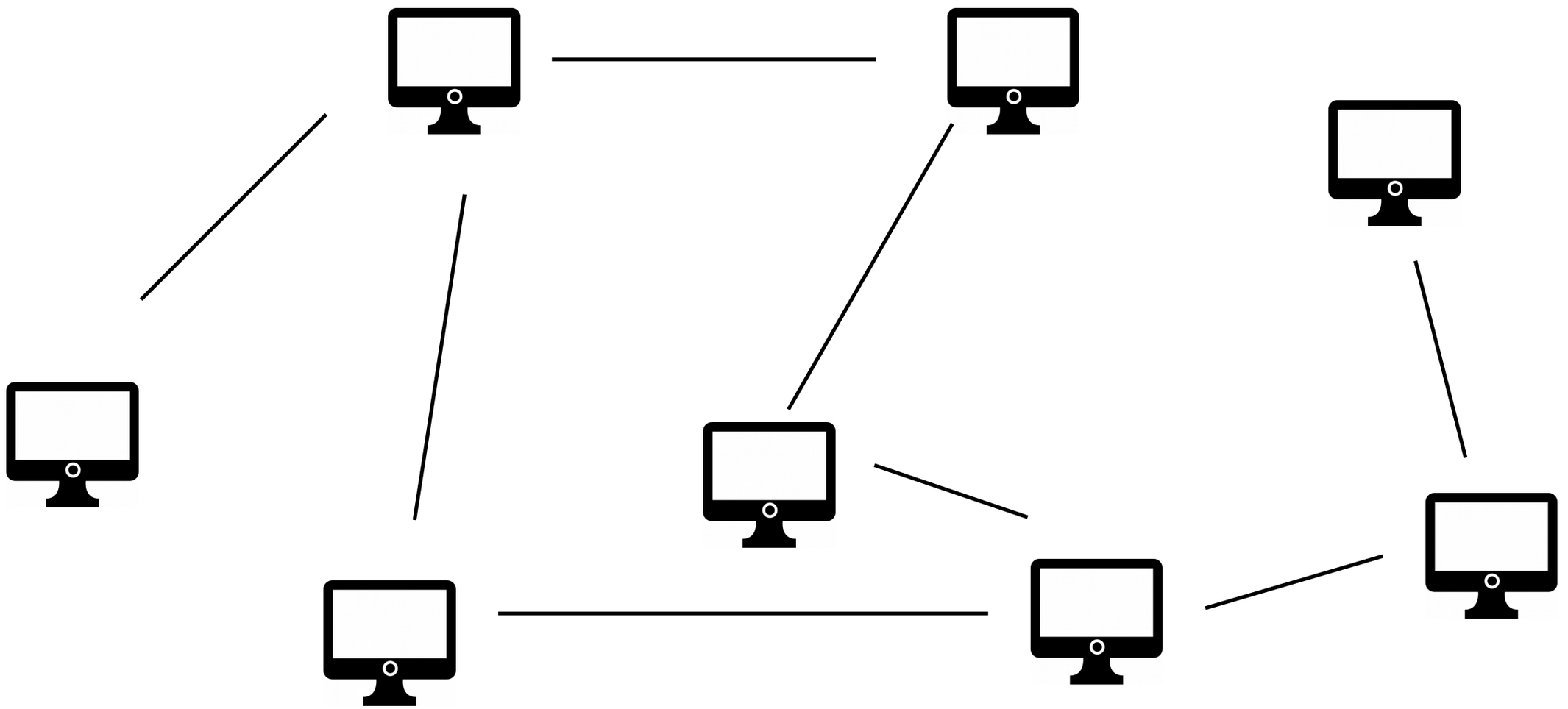
- decentralized (no single point of failure)
- guaranteed execution of the transfer
- transparency
- immutable
- potentially lower fees

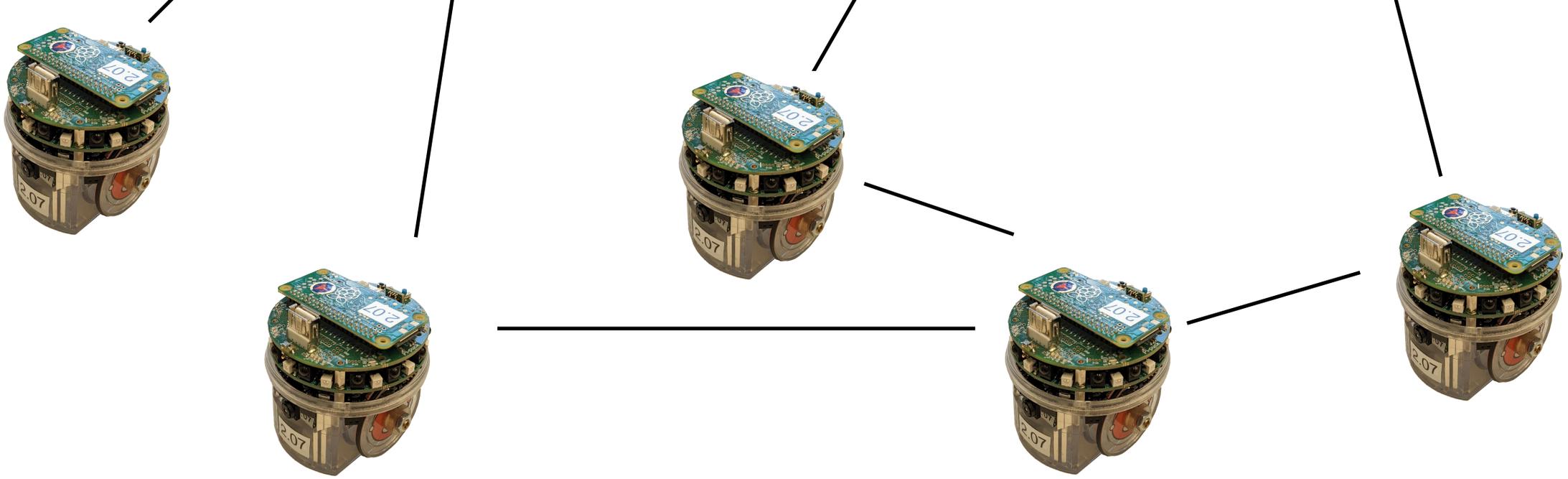
Security issues in swarm robotics

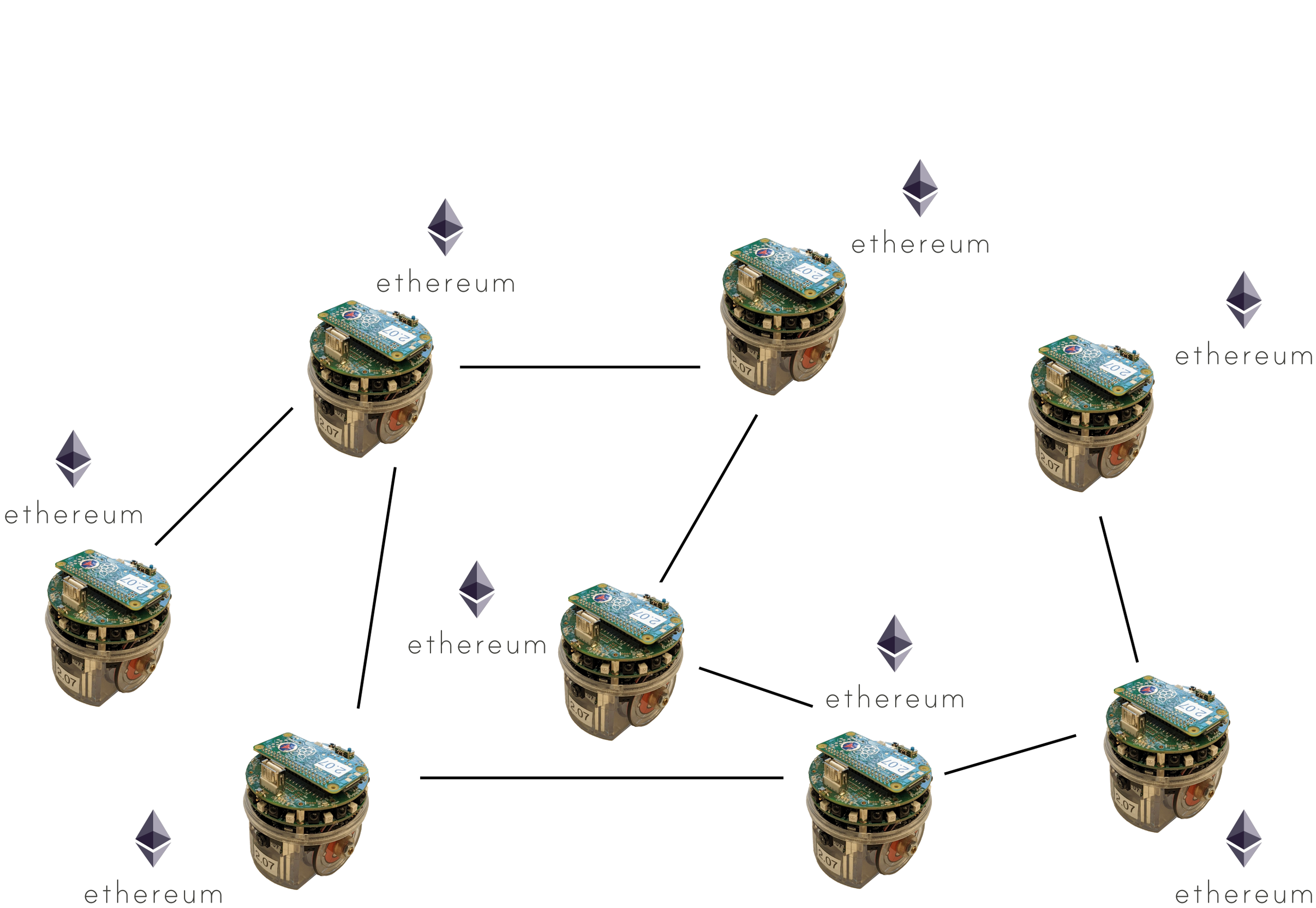
Part I
Shortcomings of classical approaches in swarm robotics

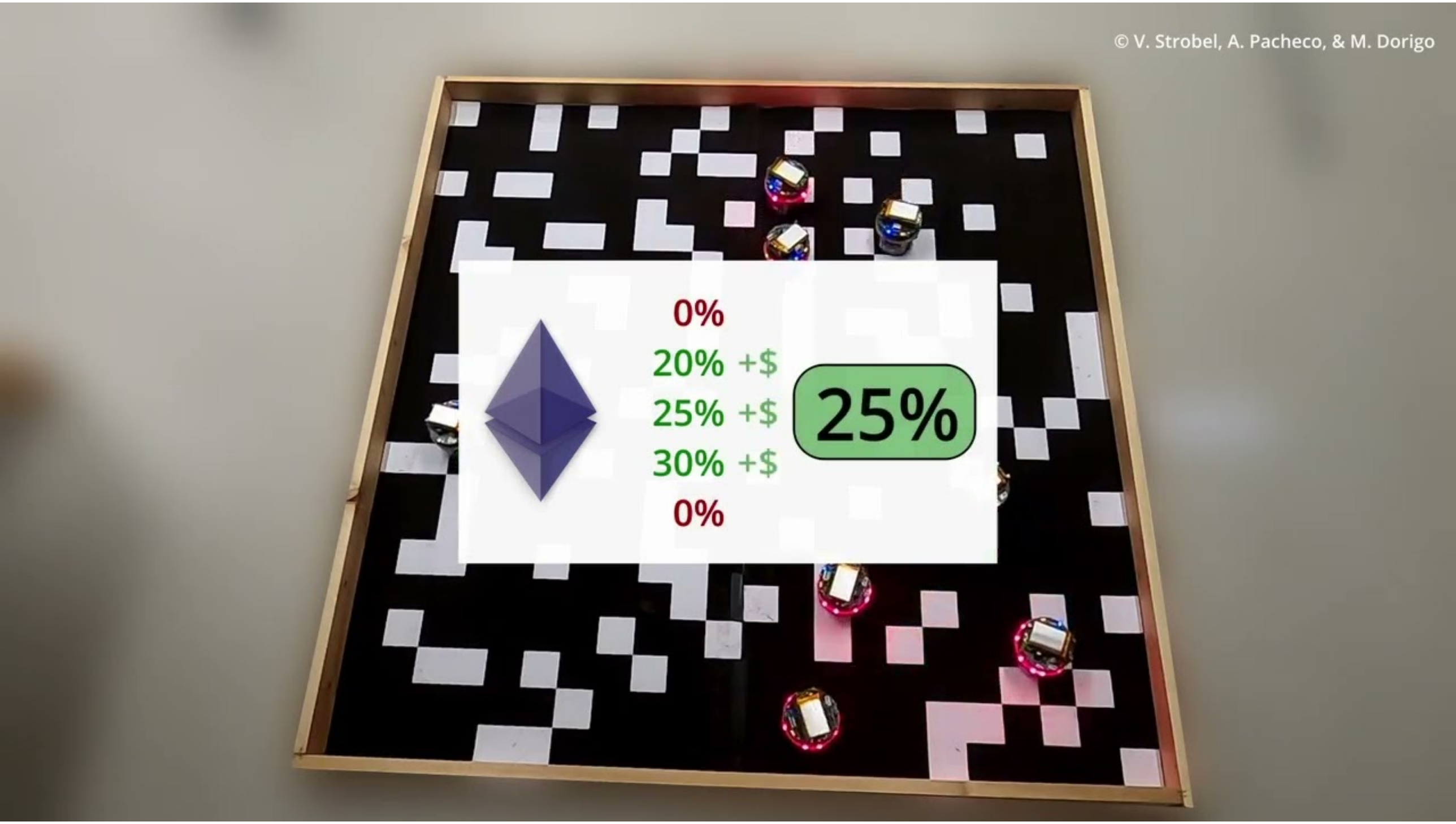
Part II
Blockchain technology

Part III
Blockchain-based smart contracts for the secure coordination of robot swarms






ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
2.07



Strobel V., A. Pacheco, & M. Dorigo (2023). Robot swarms neutralize harmful Byzantine robots using a blockchain-based token economy. Science Robotics, 8(79): eabm4636.

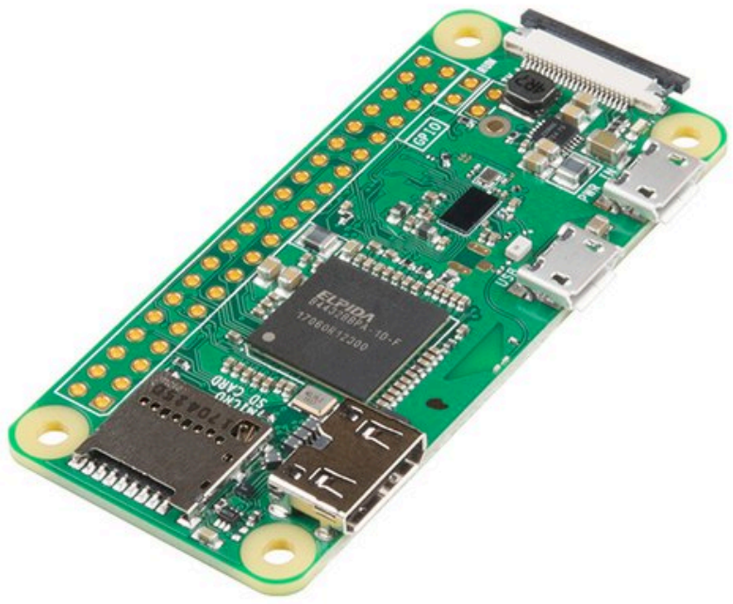
YouTube: https://www.youtube.com/watch?v=9vv0eX5-Q58

Main findings

It is possible to control a physical robot swarm via smart contracts

Message size: ~160 Bytes

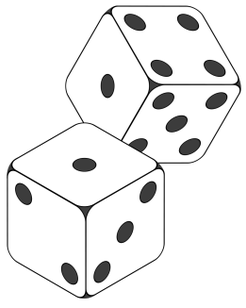
63 % black	160 B
57 % black	160 B
12 % black	160 B



Less than 35% CPU and
less than 35% RAM usage

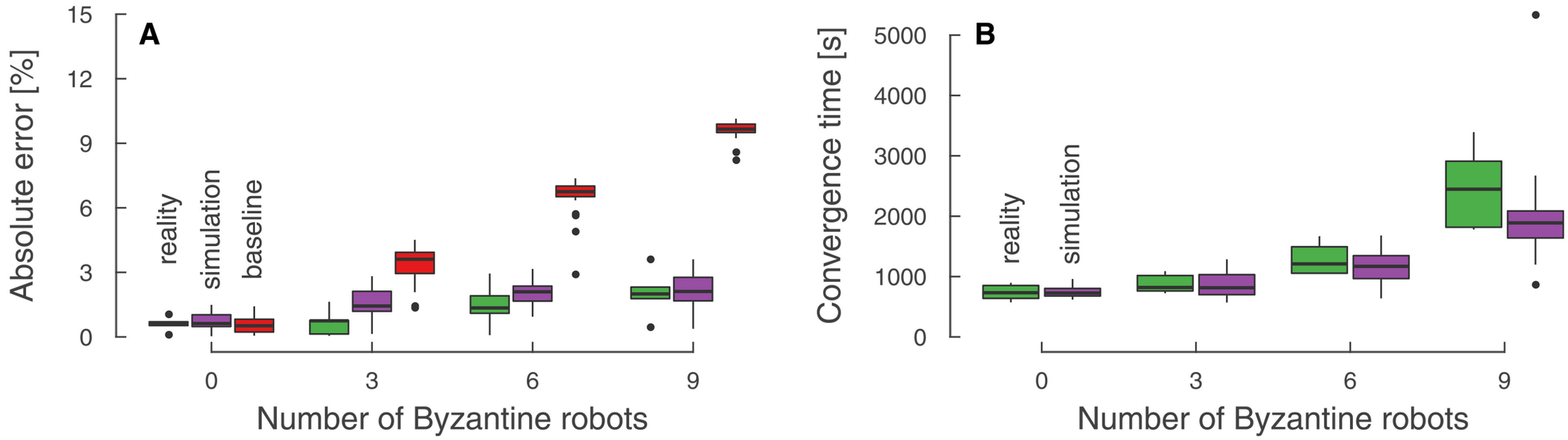
Main findings

It is possible to control a physical robot swarm via smart contracts



The smart contract is resistant to several Byzantine faults

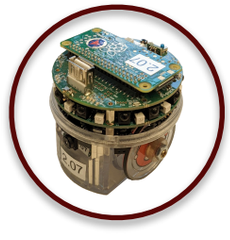
Swarm size: 24 robots



Main findings

It is possible to control a physical robot swarm via smart contracts

The smart contract is resistant to several Byzantine faults

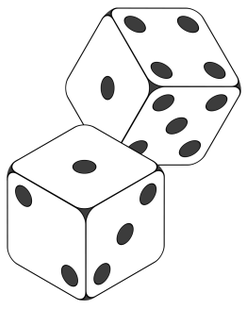


$↓

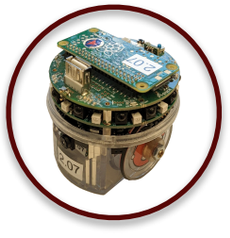
The economy is able to neutralize Byzantine robots

Main findings

It is possible to control a physical robot swarm via smart contracts

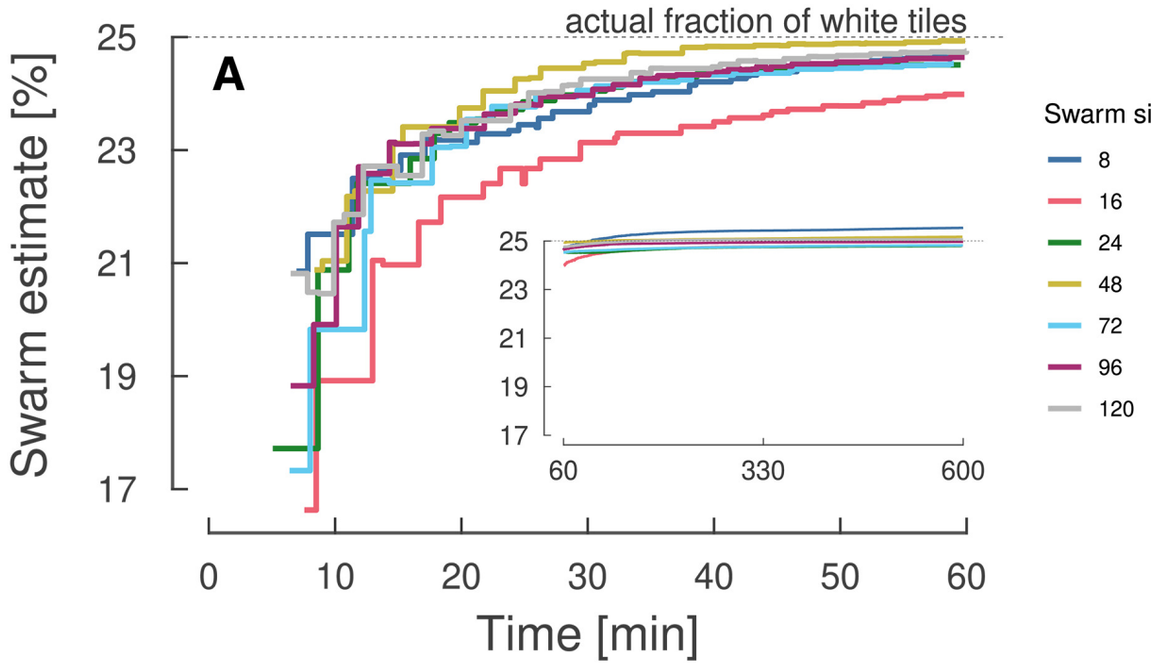


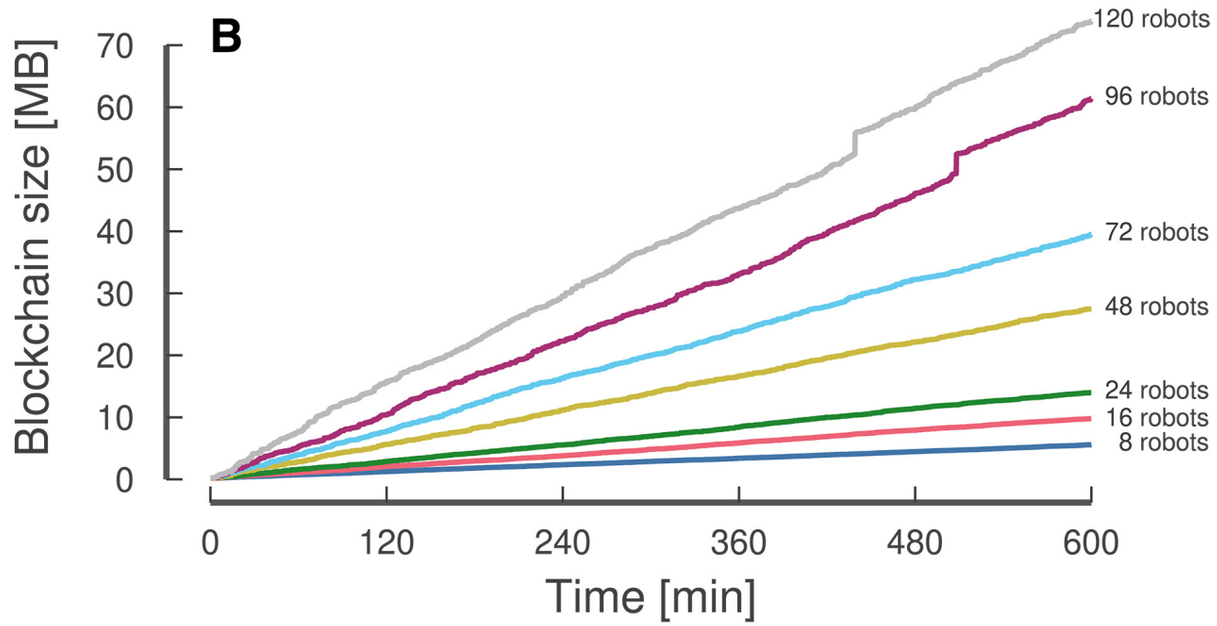
The smart contract is resistant to several Byzantine faults



The economy is able to neutralize Byzantine robots

The system is scalable


A
actual fraction of white tiles
Swarm estimate [%]
25
23
21
19
17
0
10
20
30
40
50
60
Time [min]
25
23
21
19
17
60
330
600
Swarm size
8
16
24
48
72
96
120


B
Blockchain size [MB]
70
60
50
40
30
20
10
0
0
120
240
360
480
600
Time [min]
120 robots
96 robots
72 robots
48 robots
24 robots
16 robots
8 robots

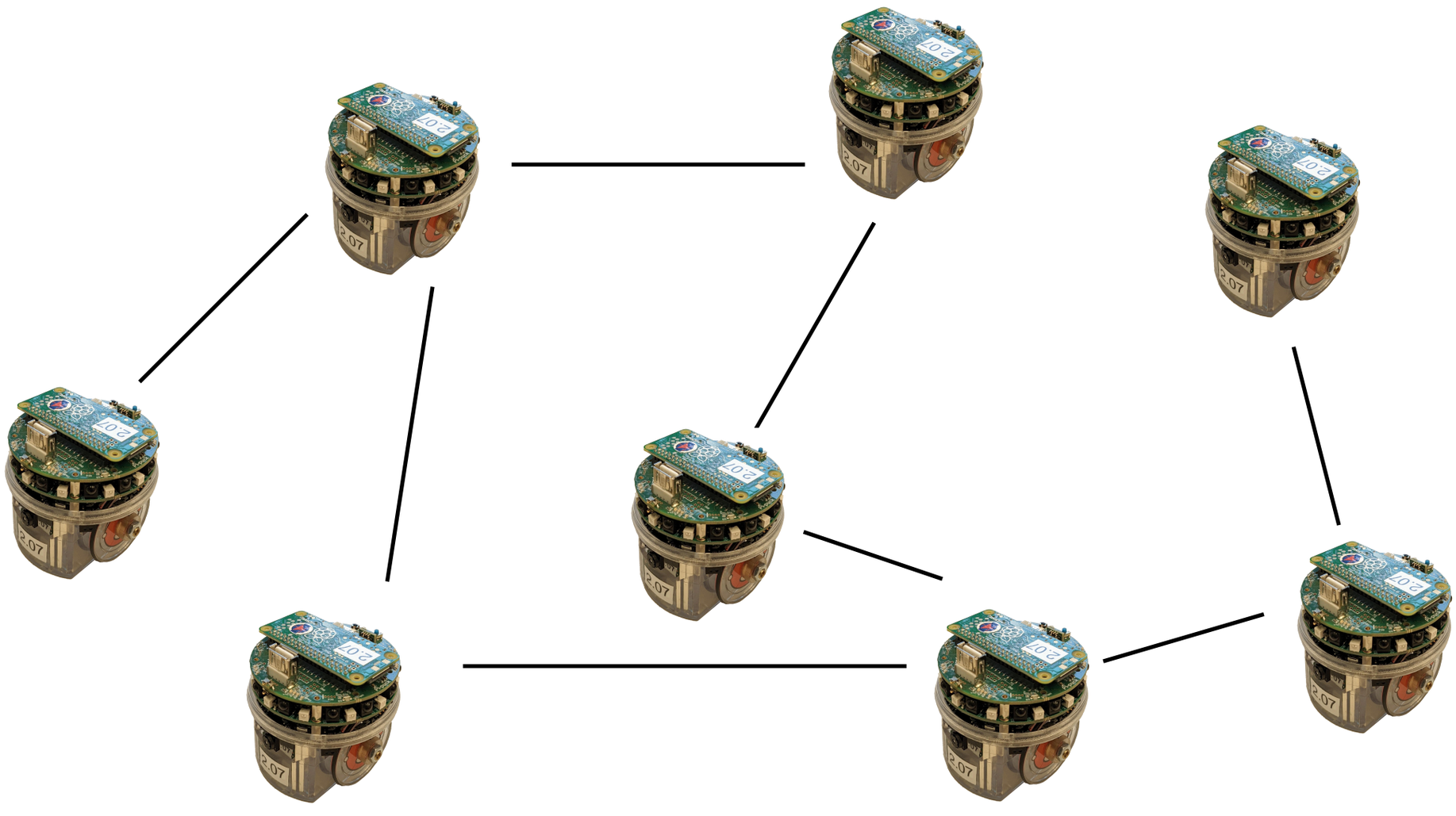
Research directions

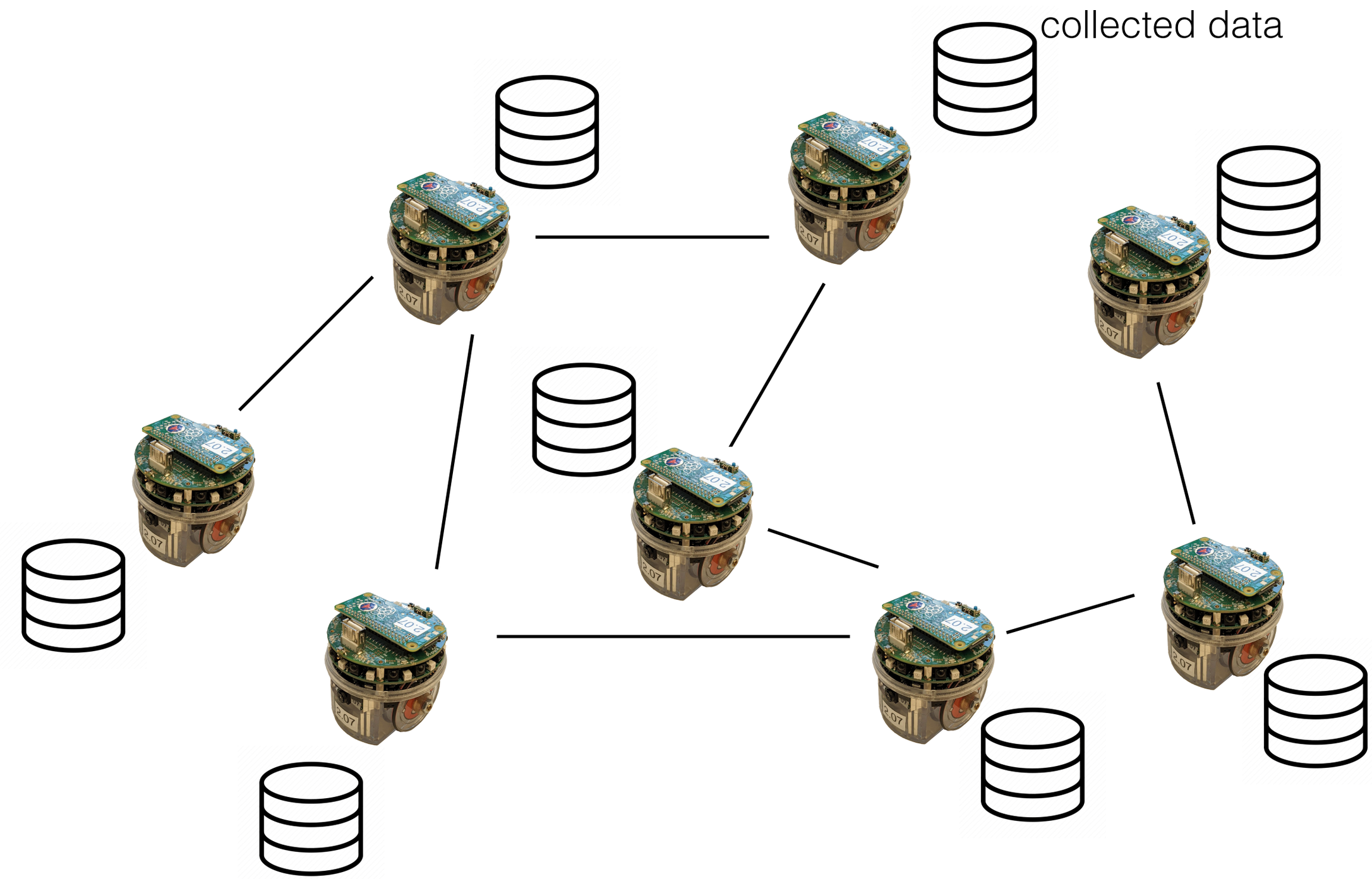
How to implement a swarm-level behavorial switch (in presence of Byzantine robots)?

How to implement secure simultaneous localization and mapping (SLAM)?

How to implement secure distributed machine learning in robot swarms?

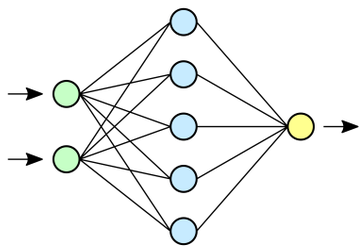
How to implement secure machine learning in robot swarms?




collected data

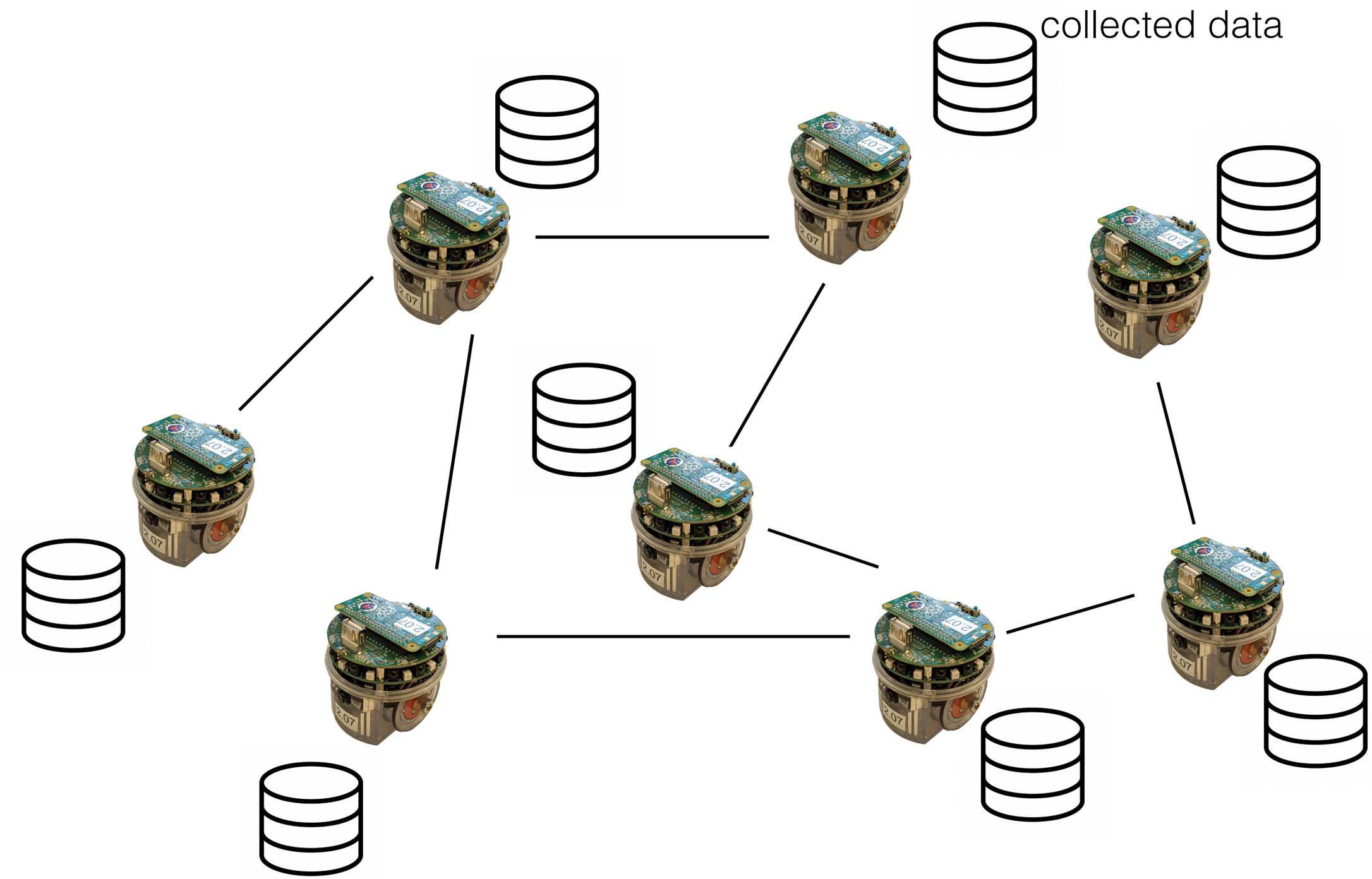
How to implement secure machine learning in robot swarms?

How to share large datasets containing confidential information?

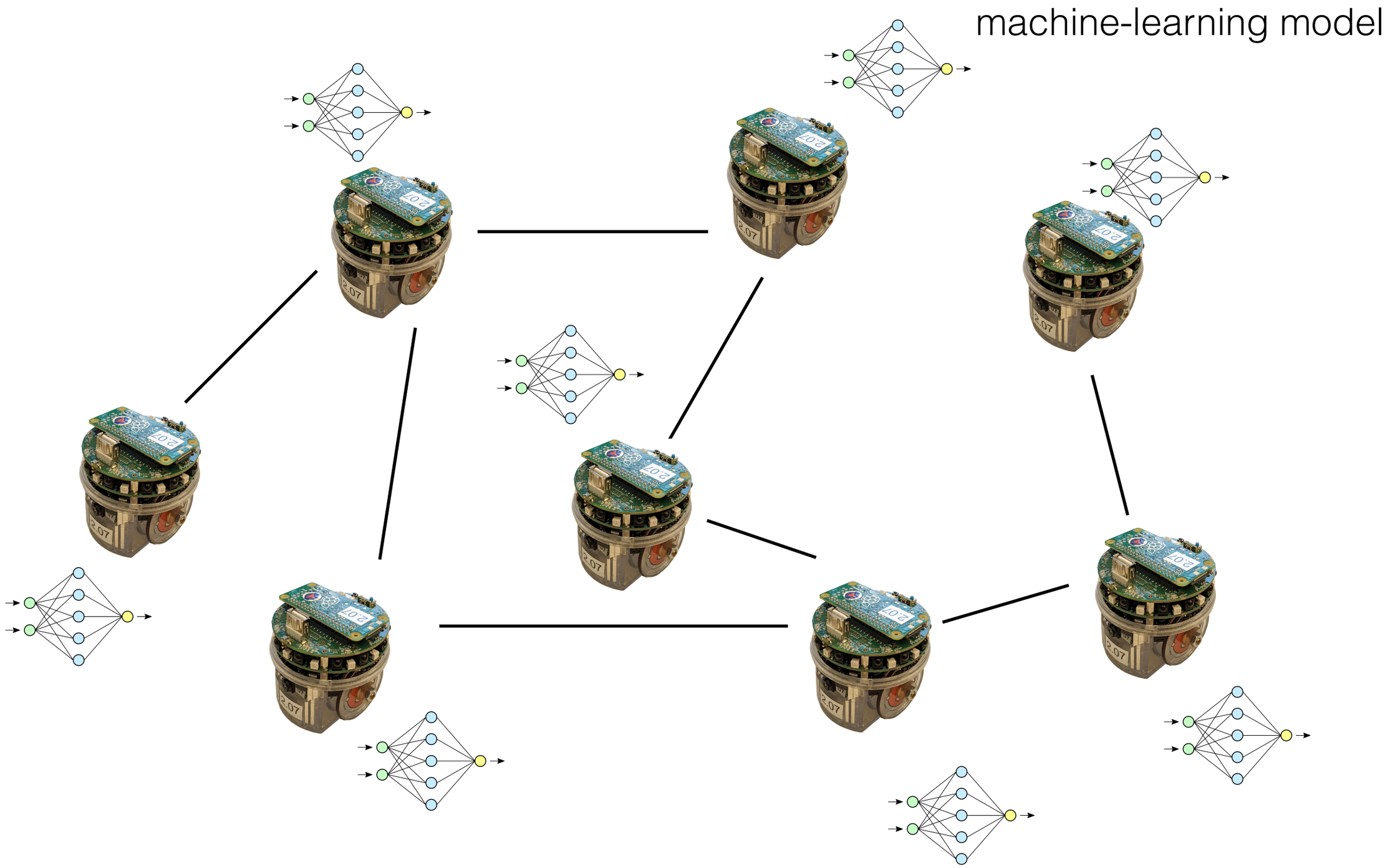


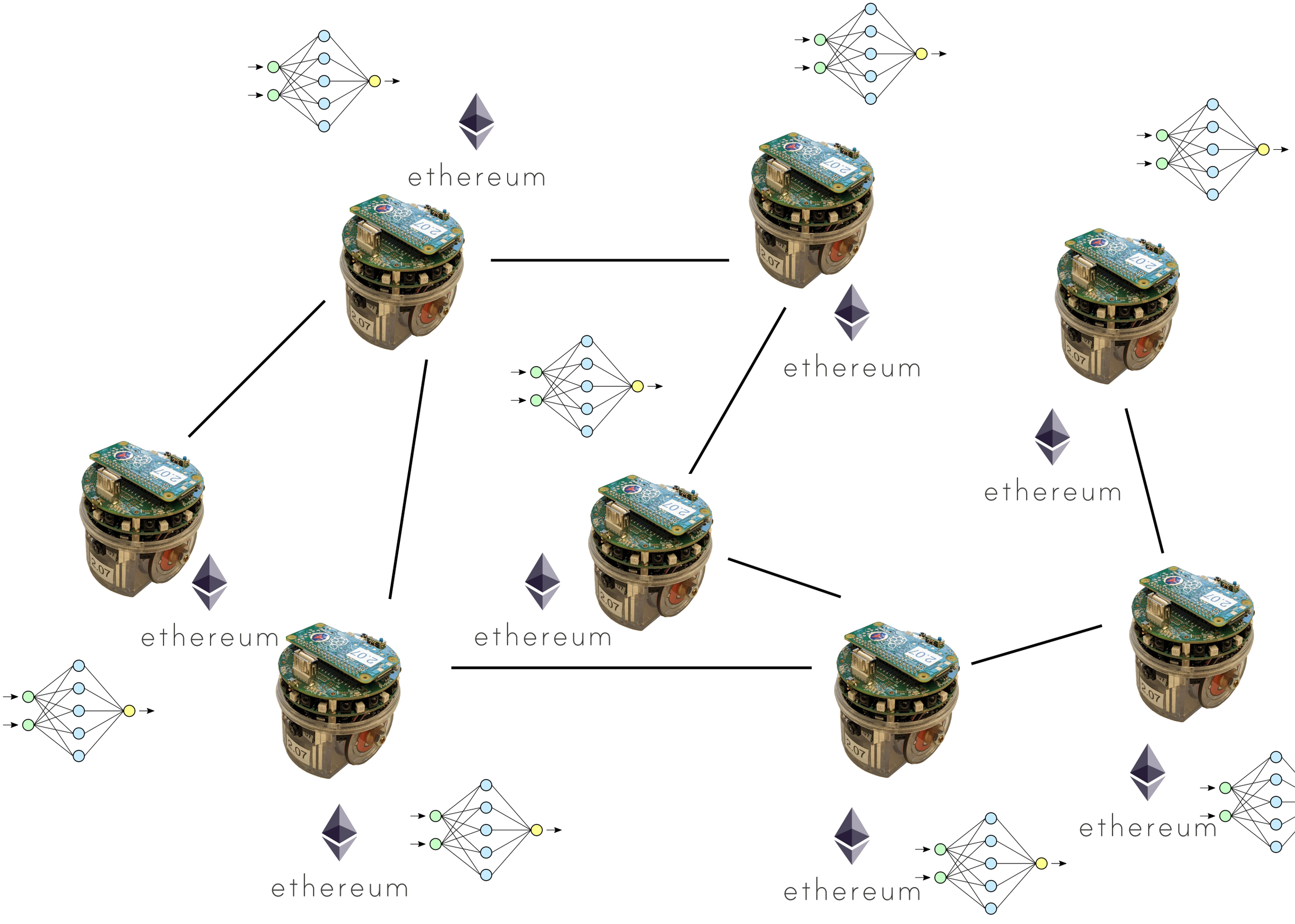
Where to train the model?

How to identify poisoning attacks?

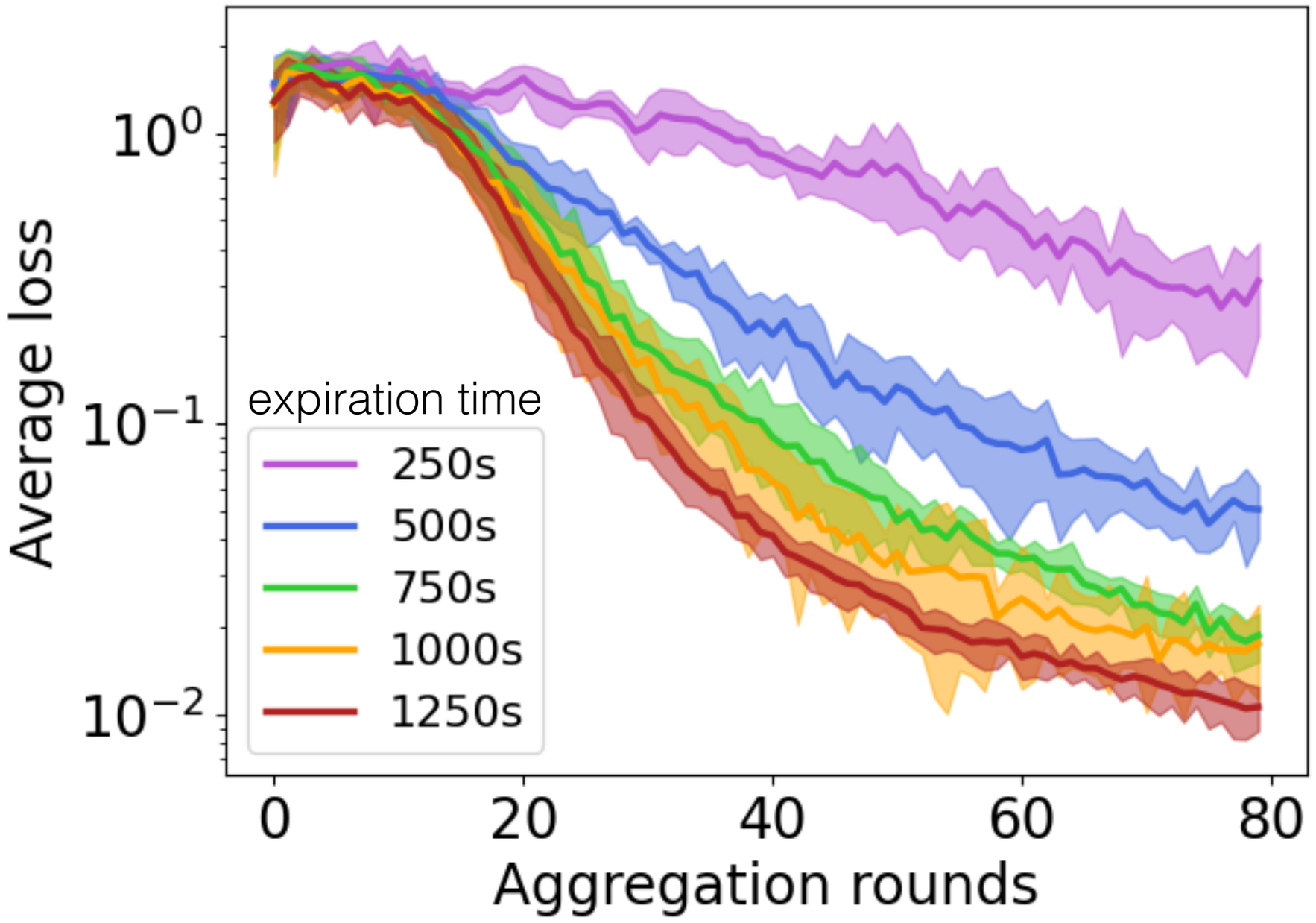

collected data

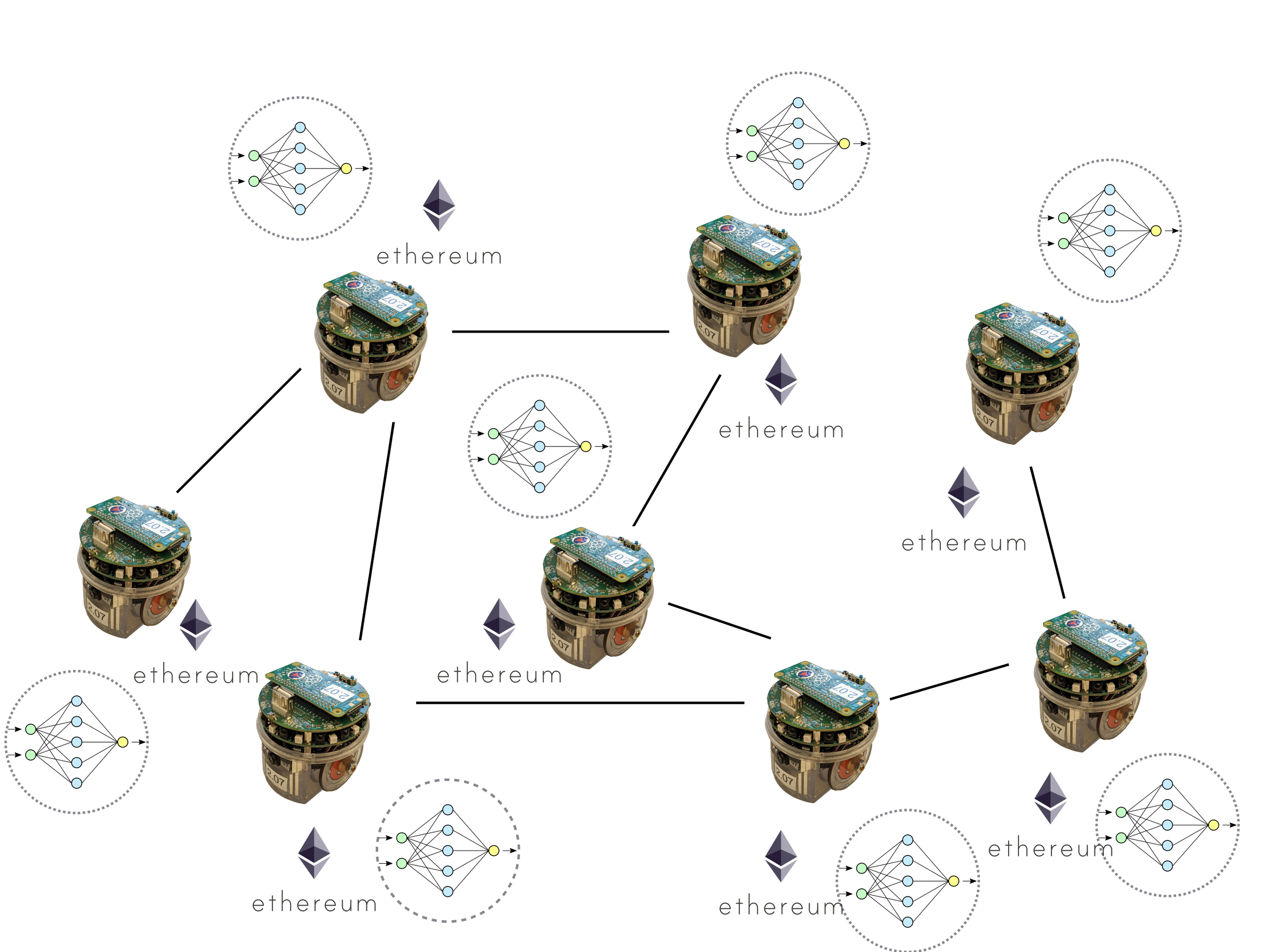
Federated learning

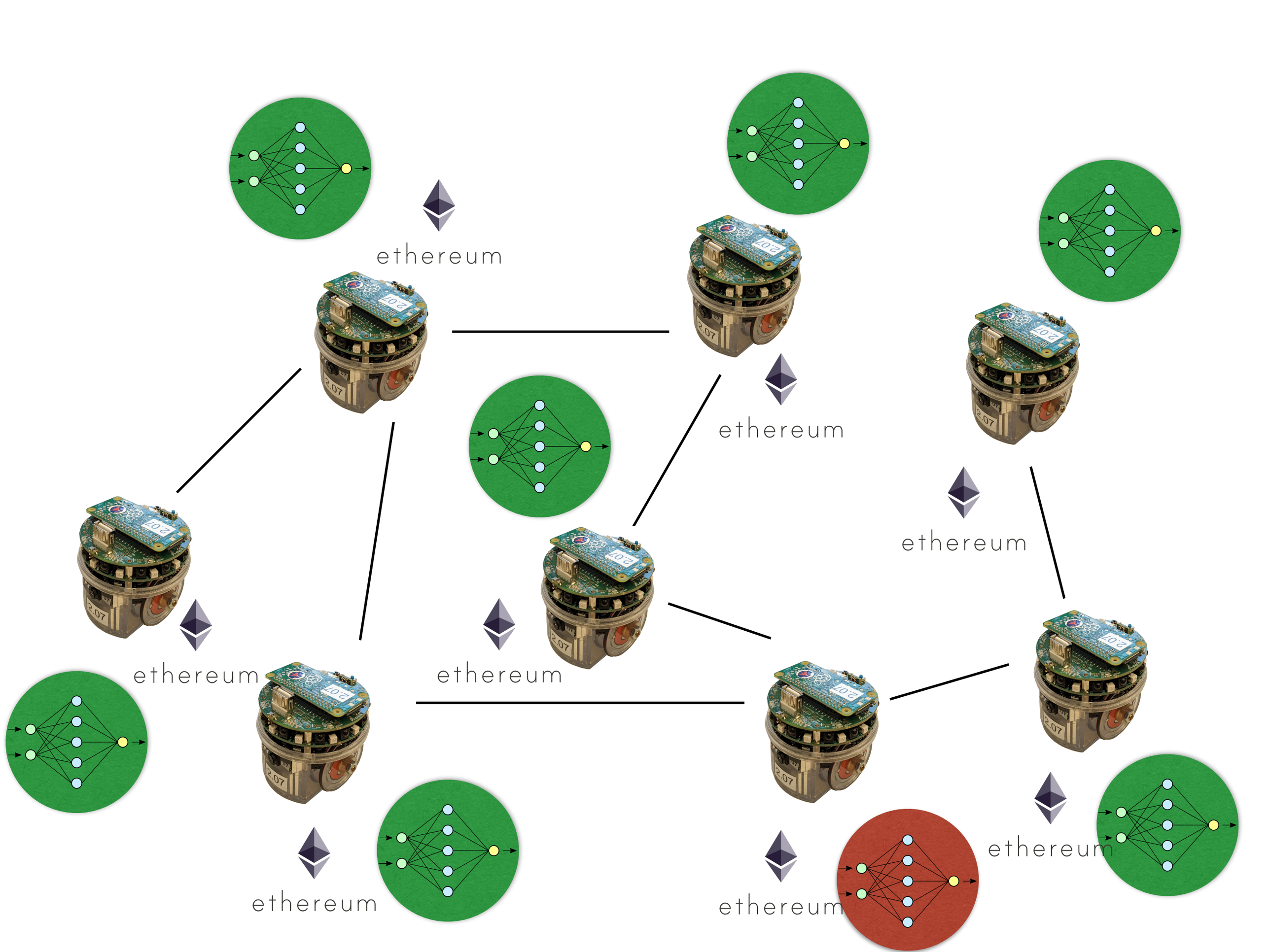



ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum

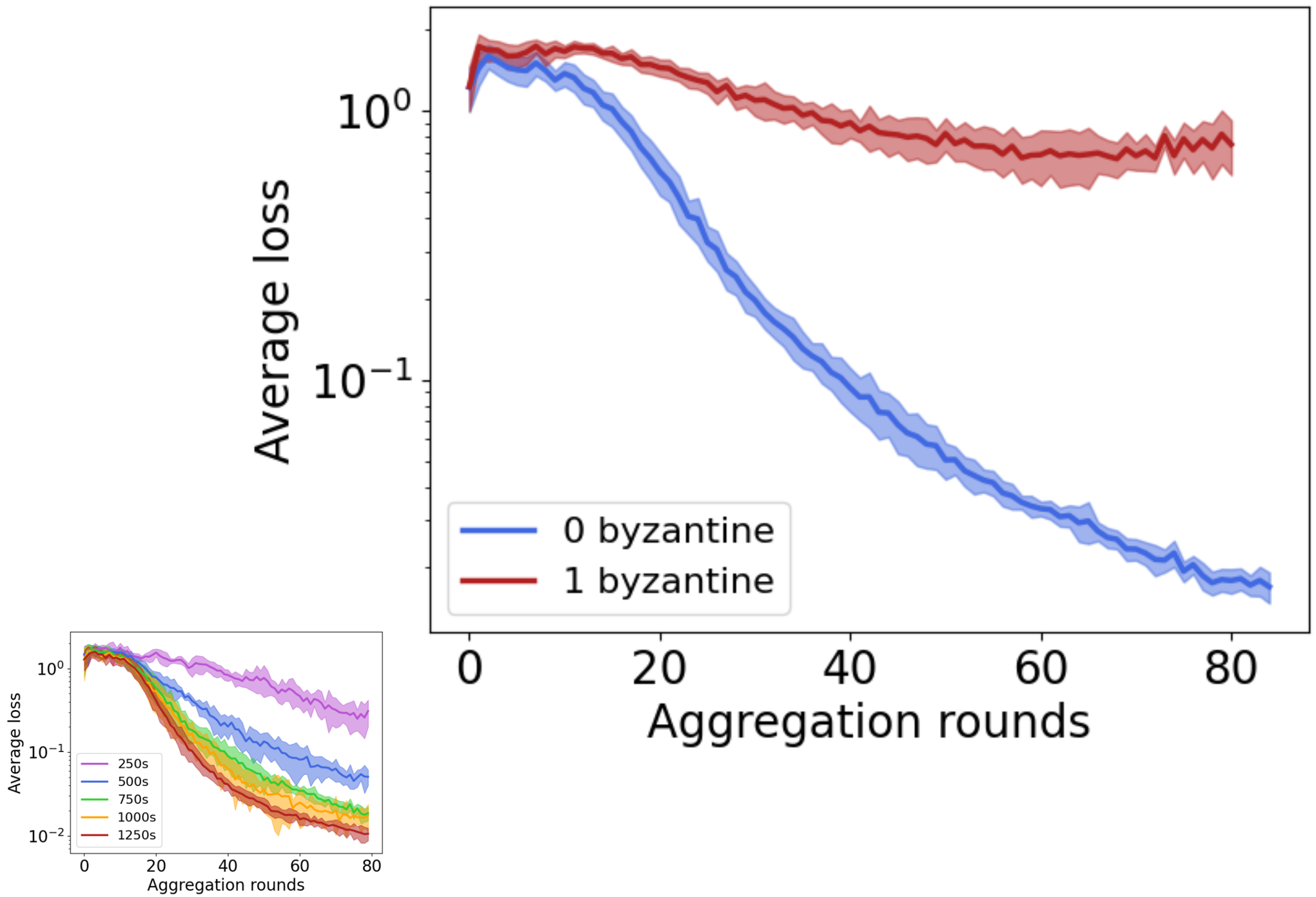
It is possible to train a machine learning model in robot swarms

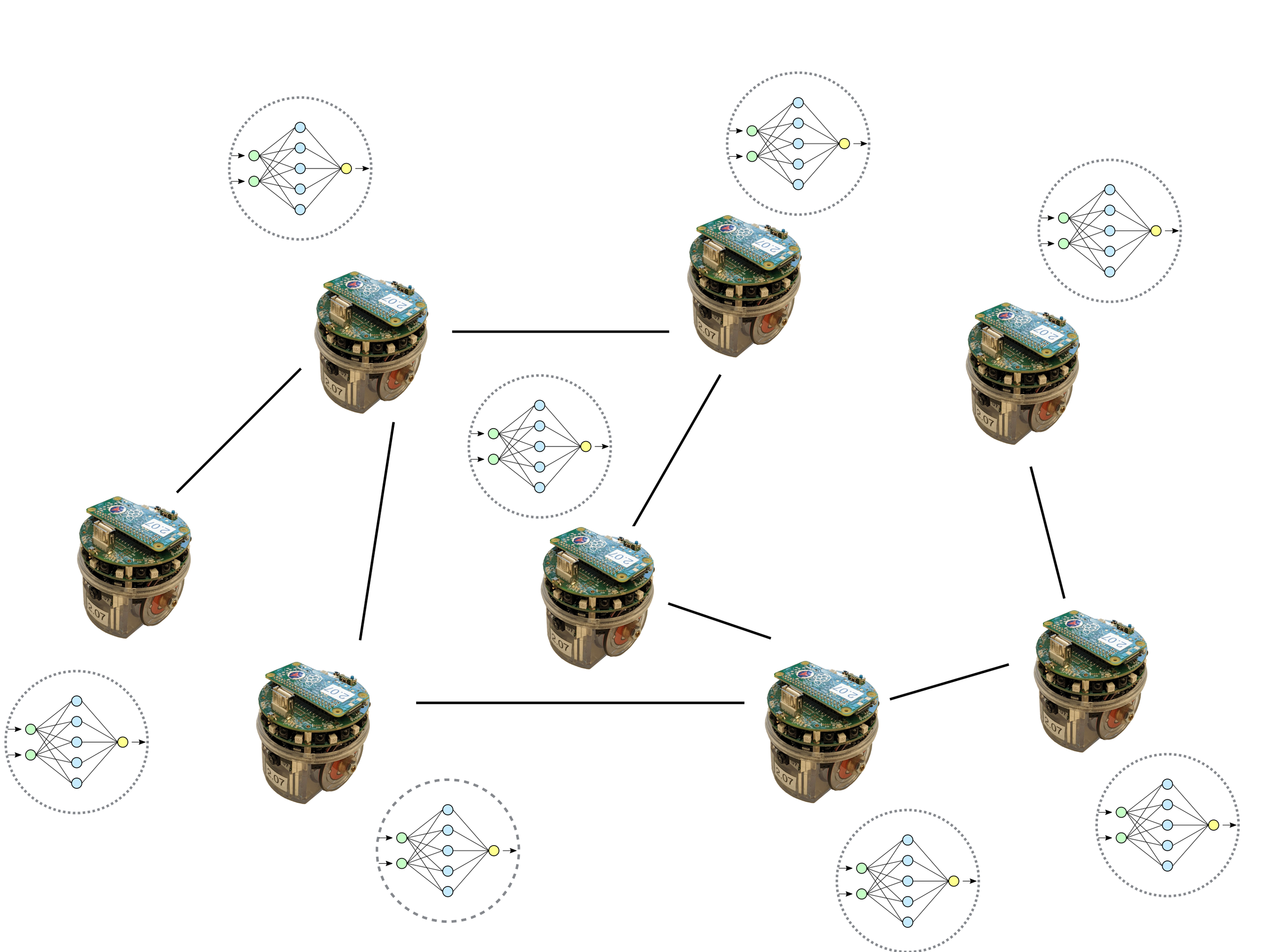



ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum

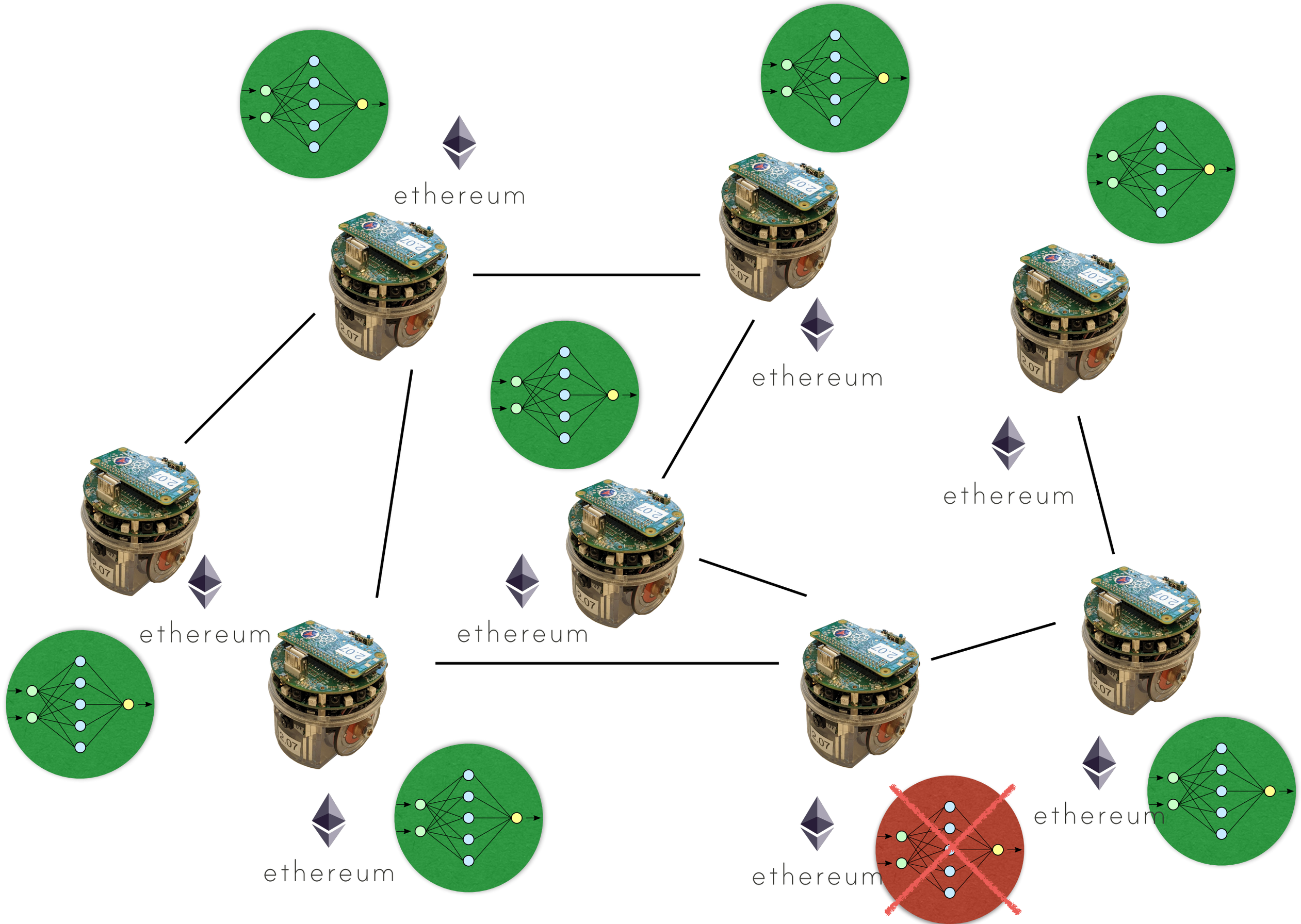

ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum
ethereum

One Byzantine robot (in a swarm of 15 robots) destroys the training

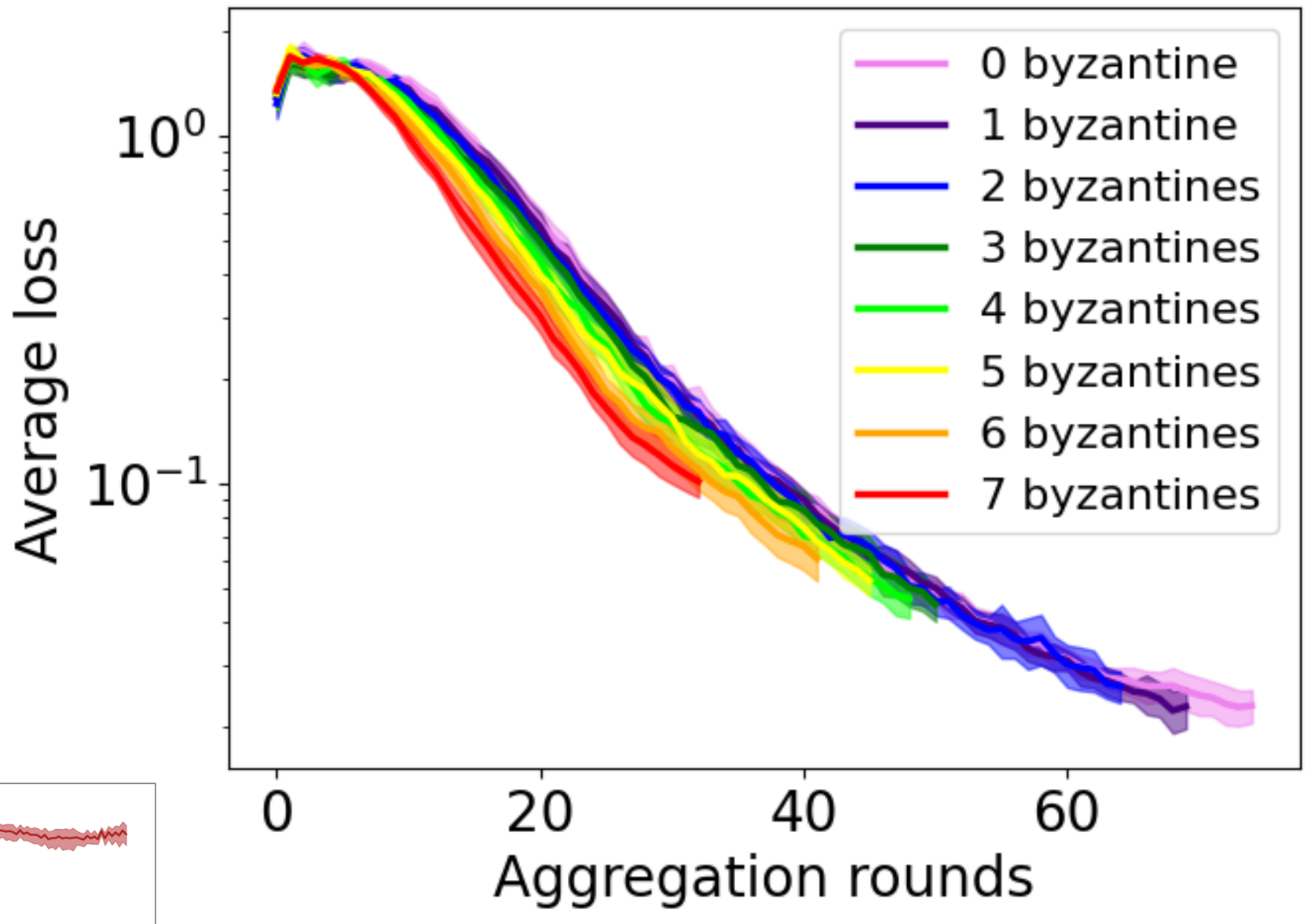


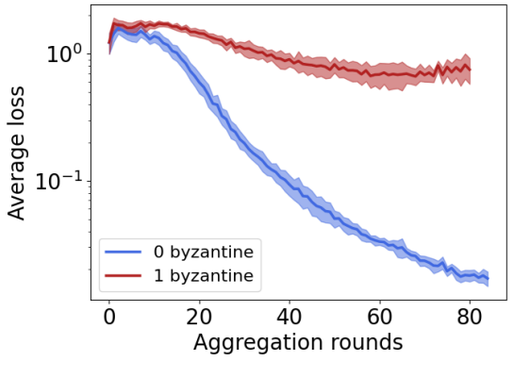


Outlier detection on the smart contract



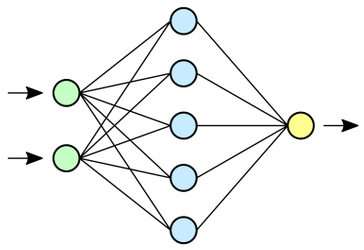
The outlier detection is able to detect and remove misleading models





How to implement secure machine learning in robot swarms?

How to share large datasets containing confidential information?
You do not

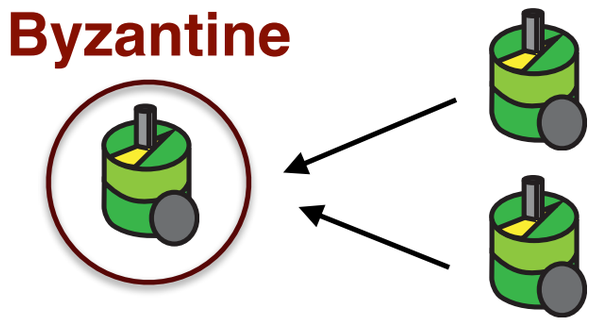


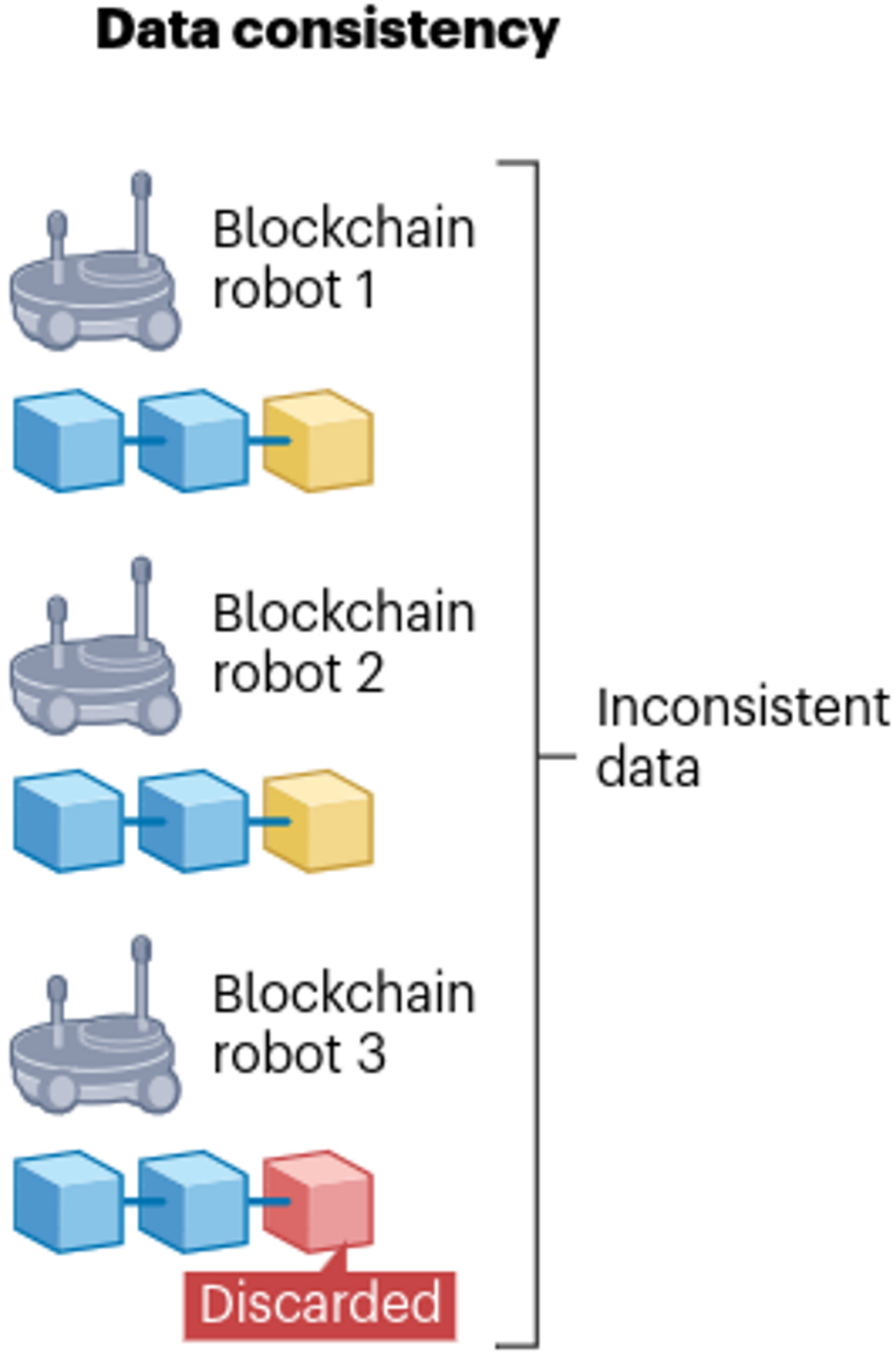
Where to train the model?
On the robot

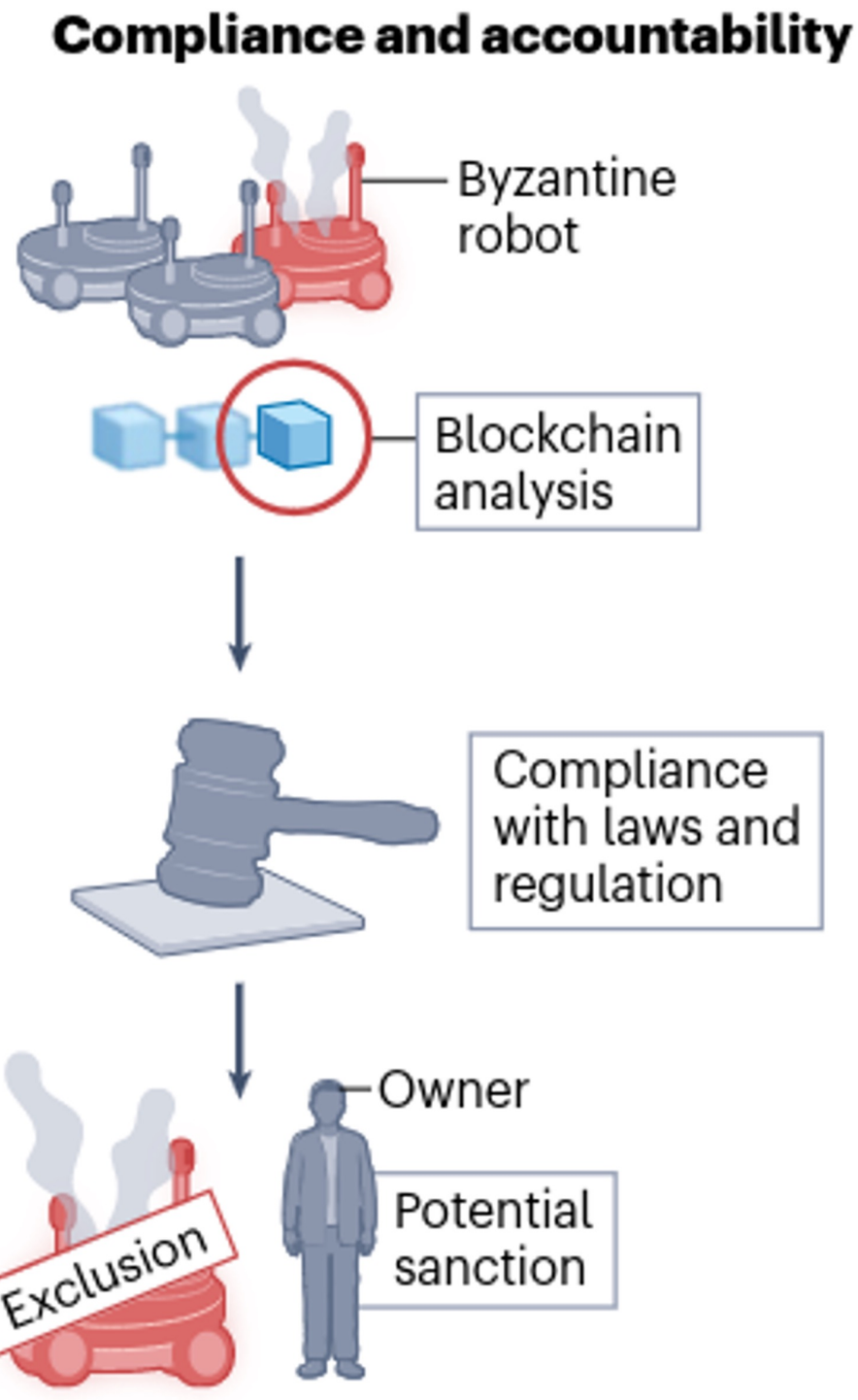
How to identify poisoning attacks?
Via the smart contract

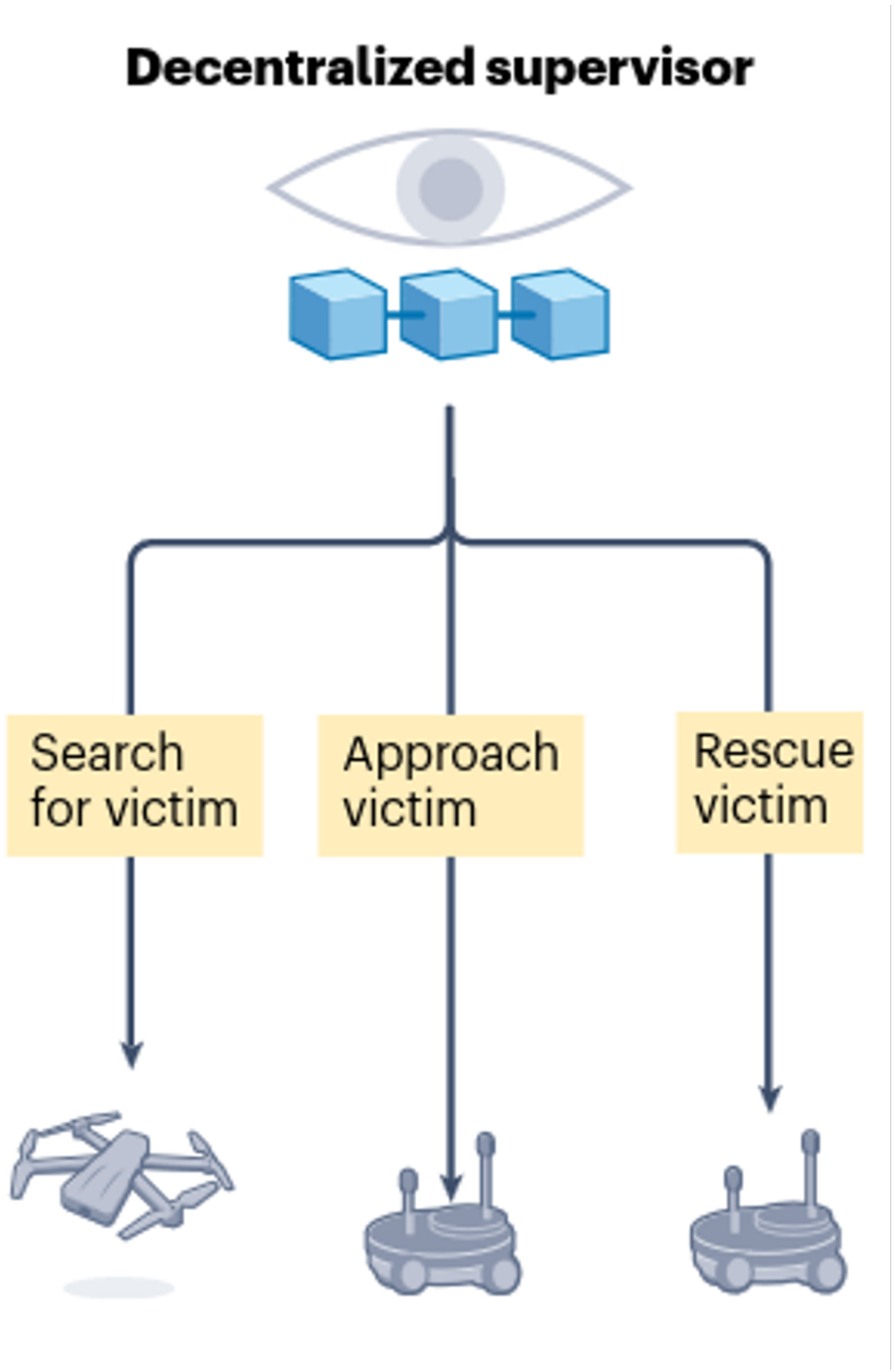
Can blockchain technology secure robot swarms against Byzantine robots?

Detect inconsistencies
and control a robot swarm



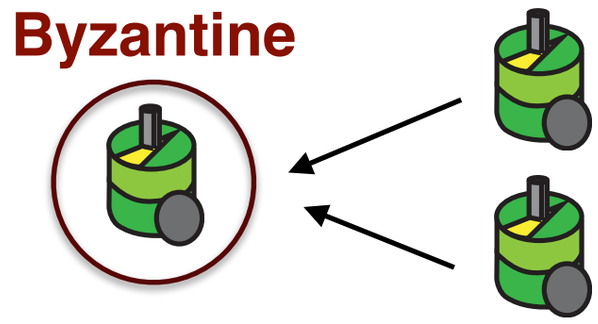

Data consistency
Blockchain robot 1
Blockchain robot 2
Blockchain robot 3
Inconsistent data
Discarded


Compliance and accountability
Byzantine robot
Blockchain analysis
Compliance with laws and regulation
Owner
Potential sanction
Exclusion

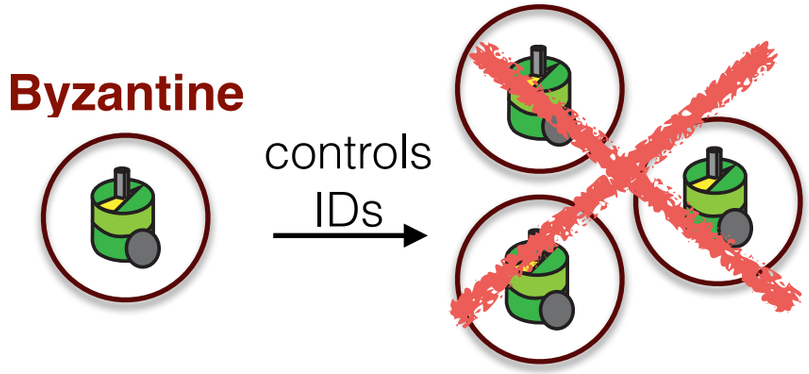

Decentralized supervisor
Search for victim
Approach victim
Rescue victim

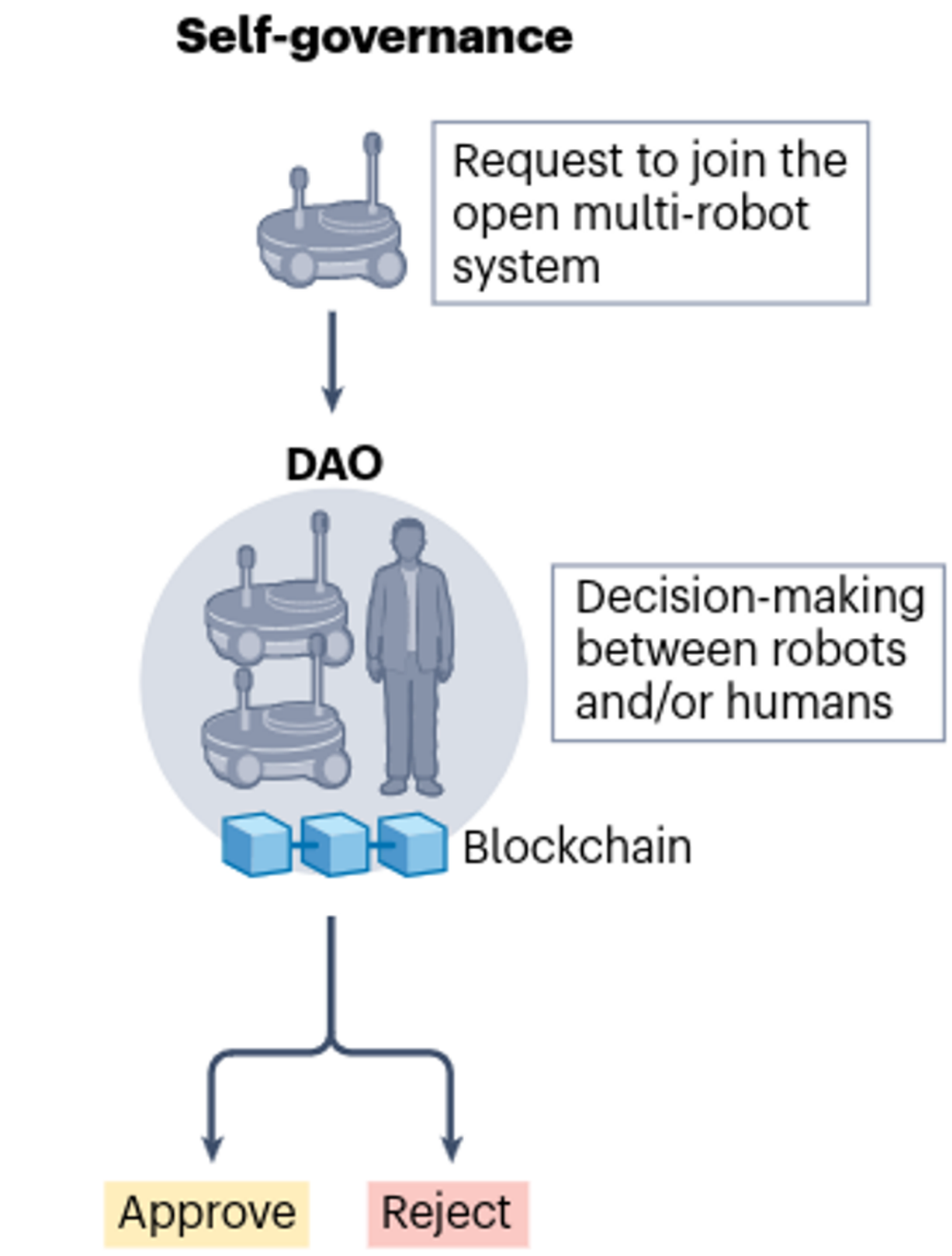
Can blockchain technology secure robot swarms against Byzantine robots?

Detect inconsistencies
and control a robot swarm



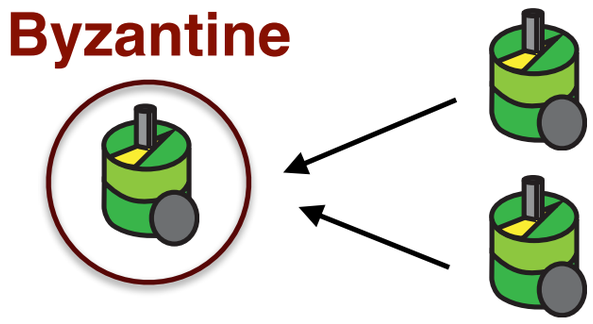
Introduce scarcity and prevent
Sybil attacks



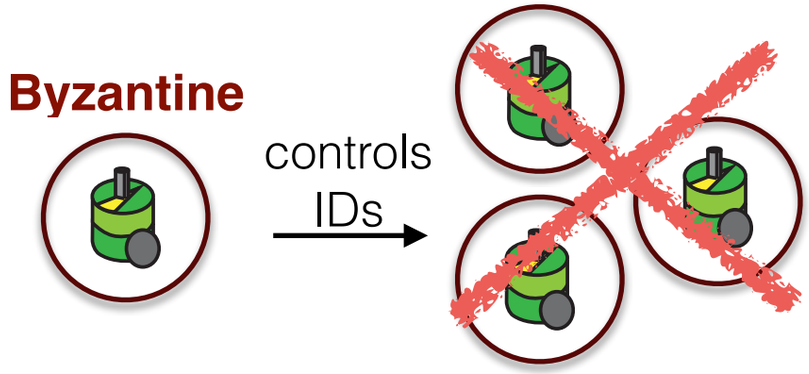

Self-governance
Request to join the open multi-robot system
DAO
Decision-making between robots and/or humans
Blockchain
Approve
Reject

Can blockchain technology secure robot swarms against Byzantine robots?

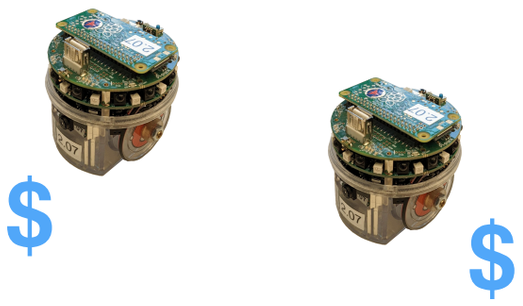
Detect inconsistencies
and and control a robot swarm



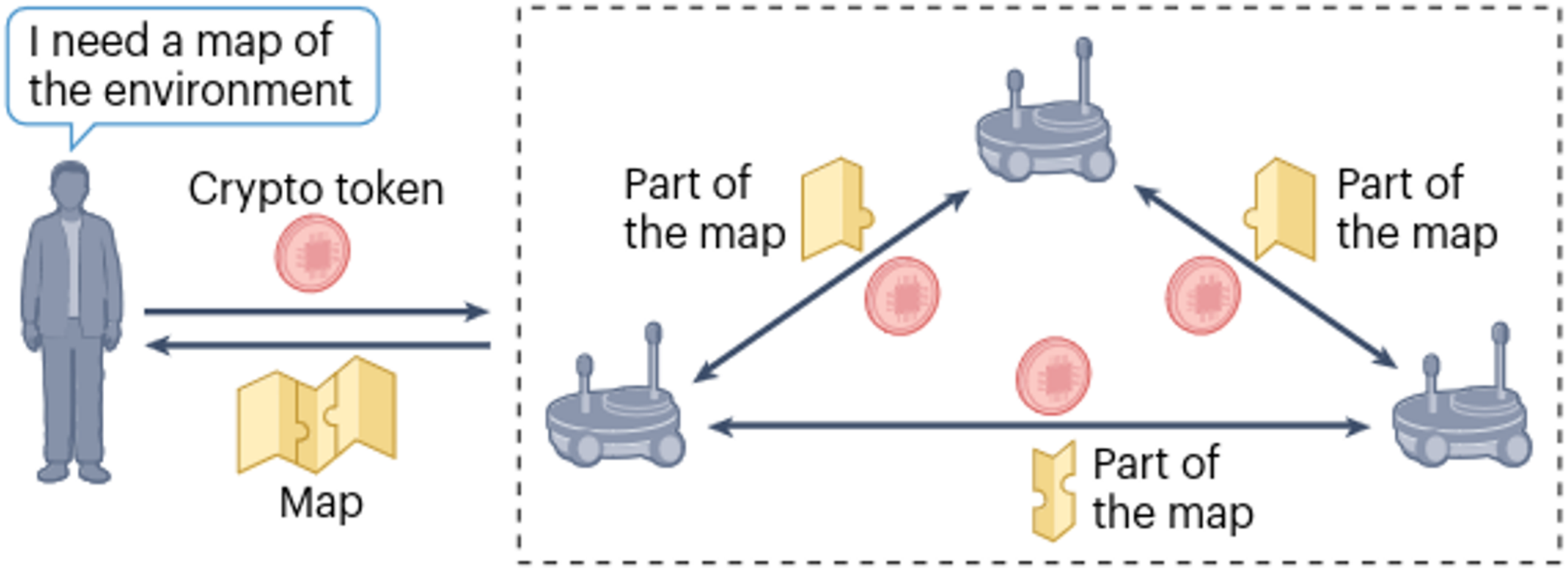
Introduce scarcity and prevent
Sybil attacks

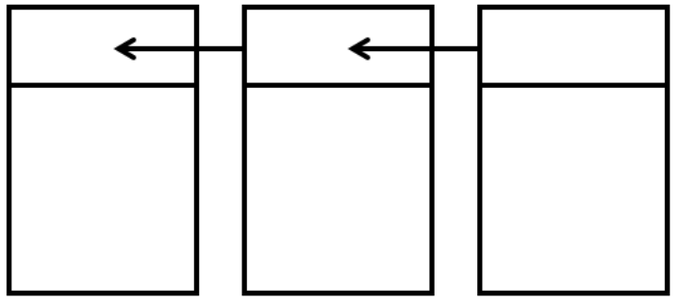


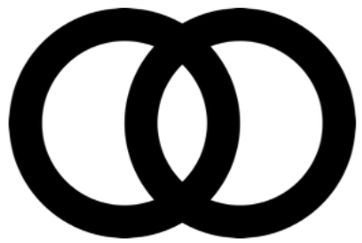
Create an economy among robots



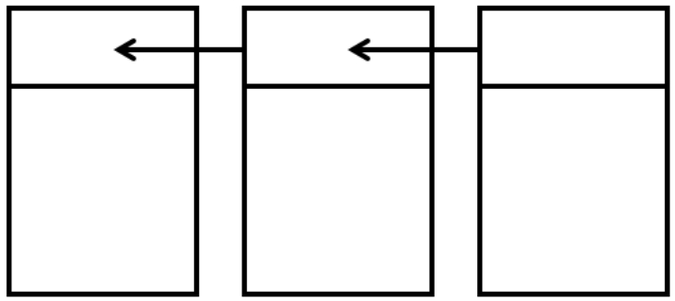
Robot economy

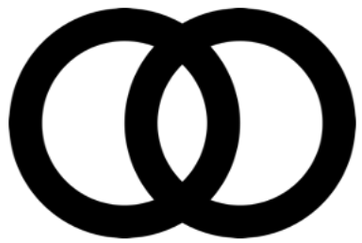






In order to identify Byzantine robots,
you need a secure system.





In order to identify Byzantine robots,
you need a secure system.

Once you have a secure system, you can do much more than identifying Byzantine robots.

Thank you!

Volker Strobel

volker.strobel@ulb.be