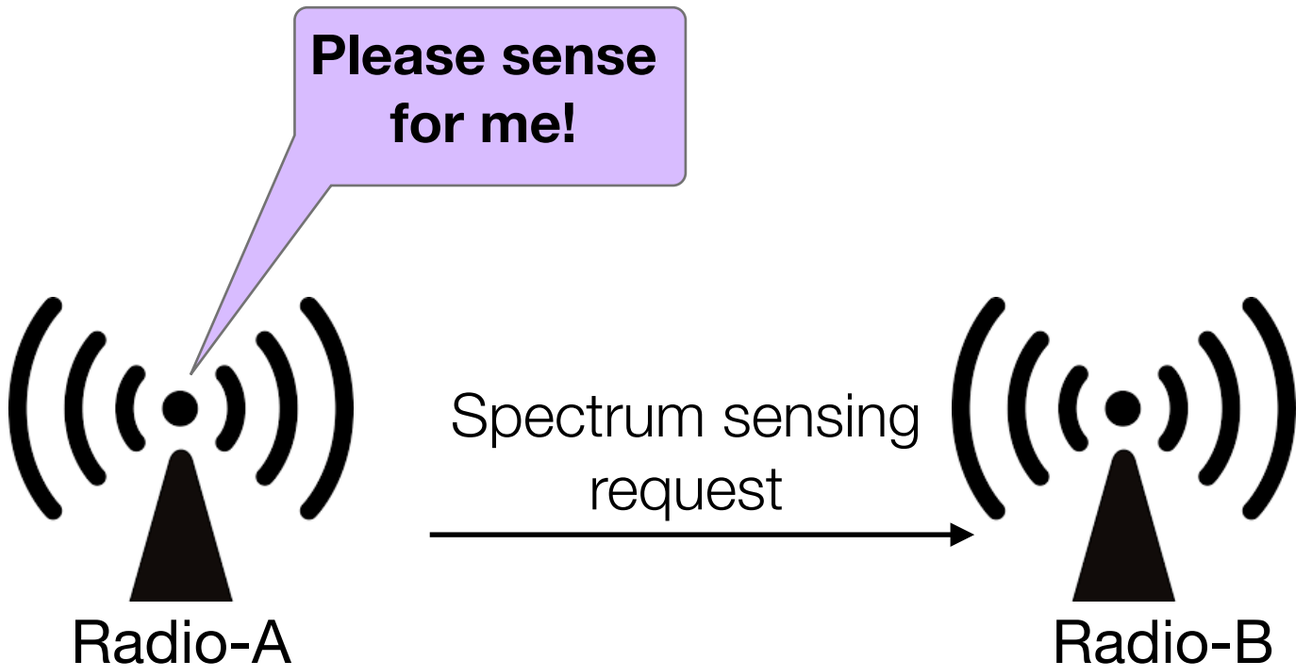
Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts

* *spass* means *fun* in German

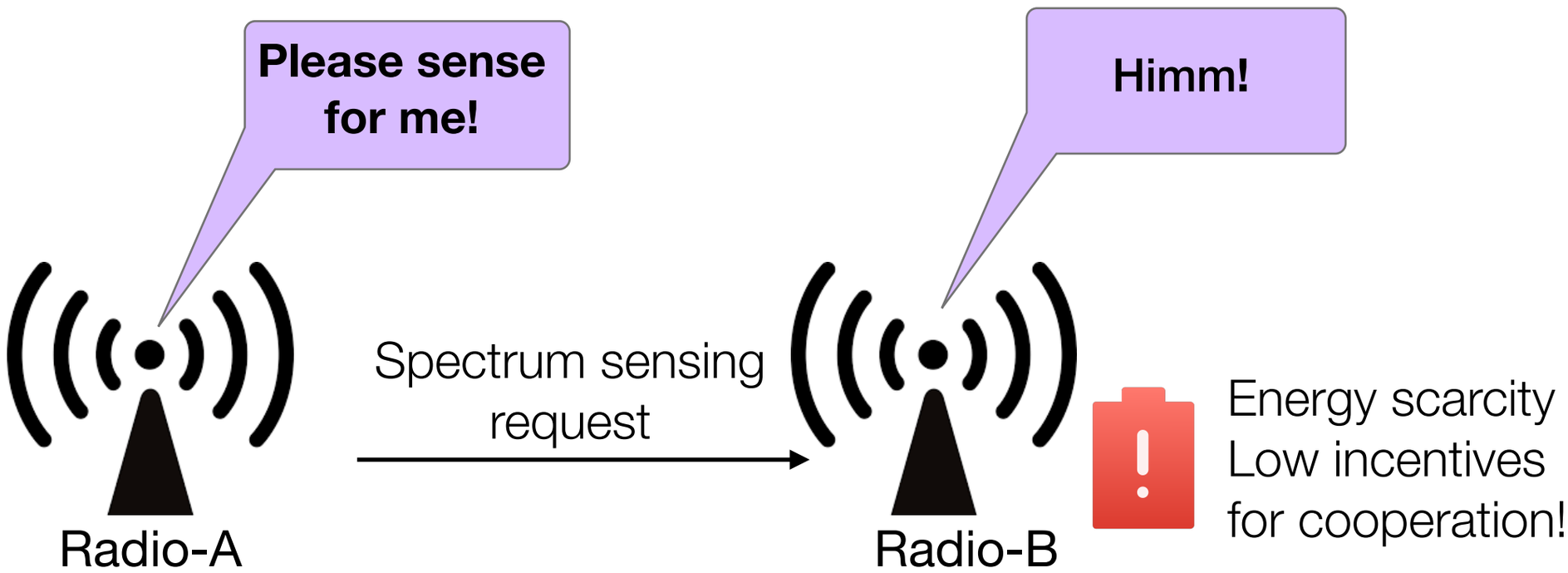
Suzan Bayhan, Anatolij Zubow, and Adam Wolisz
TU Berlin, Germany
https://suzanbayhan.github.io/

IEEE DYSPAN 2018, Seoul, South Korea
October 21-25, 2018

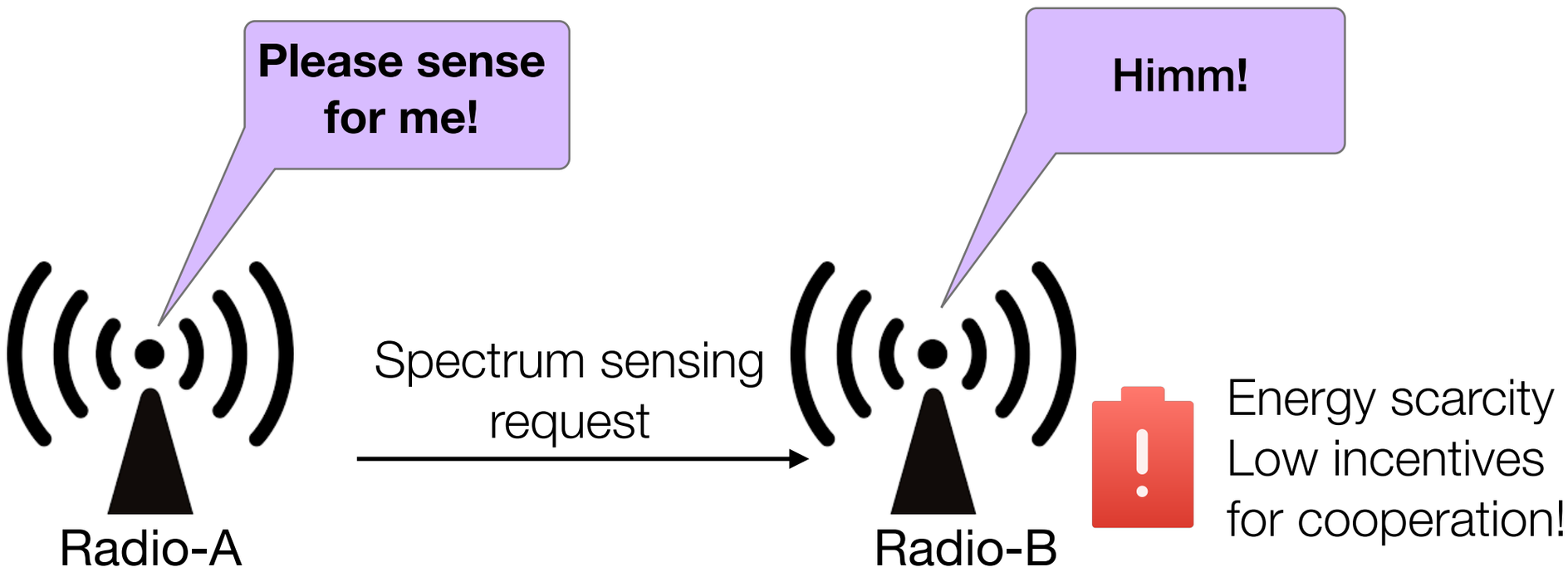
Dynamic spectrum access:
a promising solution to spectrum scarcity



Dynamic spectrum access:
a promising solution to spectrum scarcity



Dynamic spectrum access:
a promising solution to spectrum scarcity



Why to sense for others?

Why to sense for others?

- I sense whoever you are for the good of the universe
- Sense for me, I will sense for you
- I am sensing for friends

Almost all cooperative spectrum sensing literature

Reciprocity based sensing

Social-aware sensing

Why to sense for others?

- I sense whoever you are for the good of the universe
- Sense for me, I will sense for you
- I am sensing for friends

Almost all cooperative spectrum sensing literature

Reciprocity based sensing

Social-aware sensing

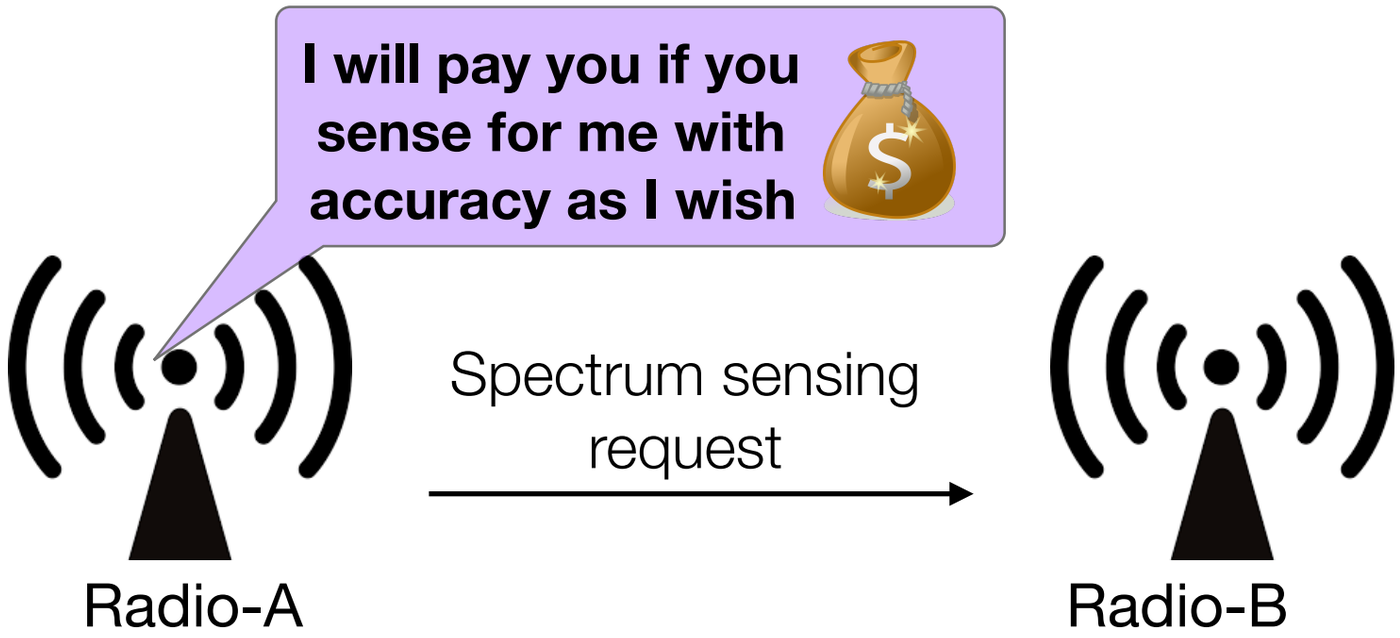
Low feasibility as business models

Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts

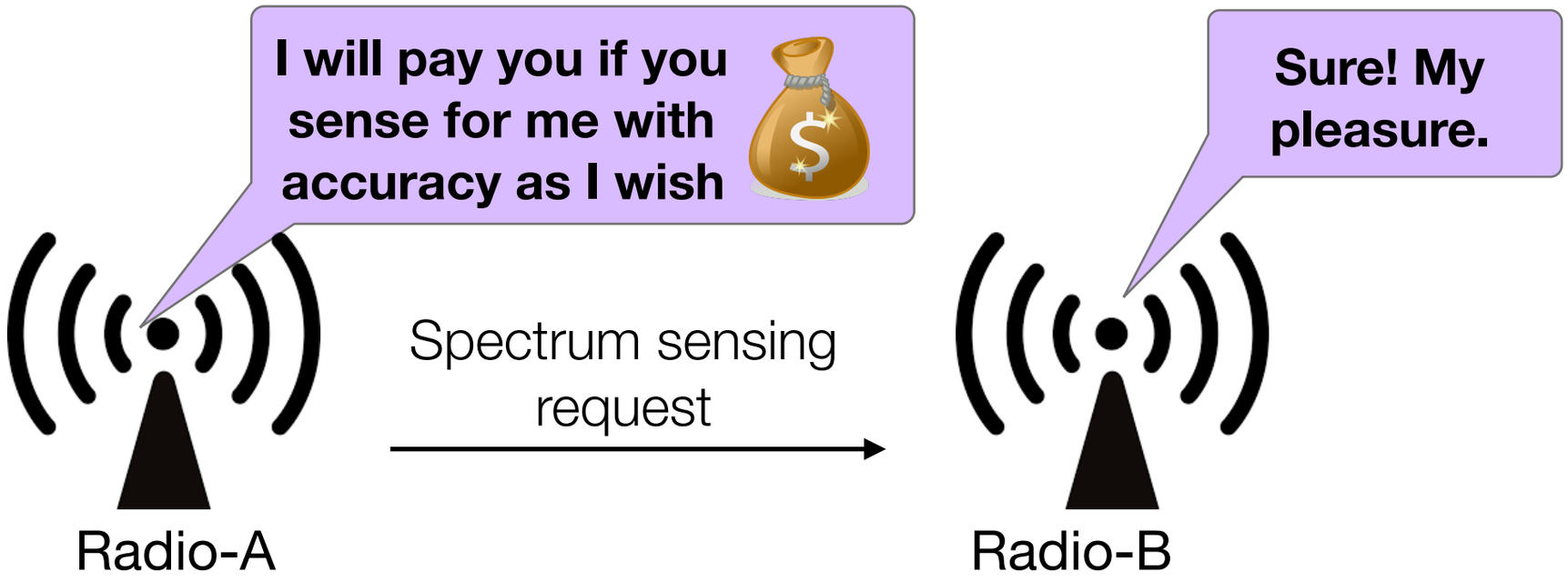
Radio-A

Radio-B

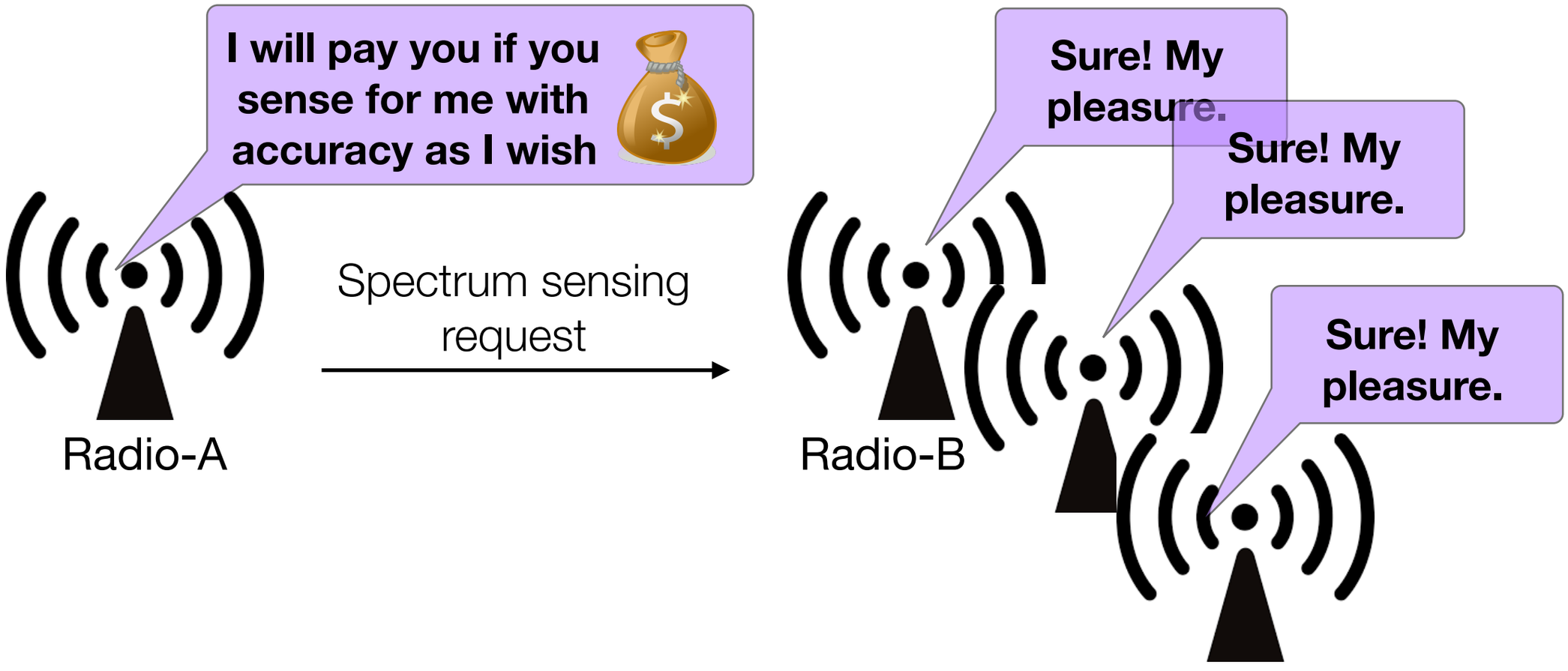
Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts



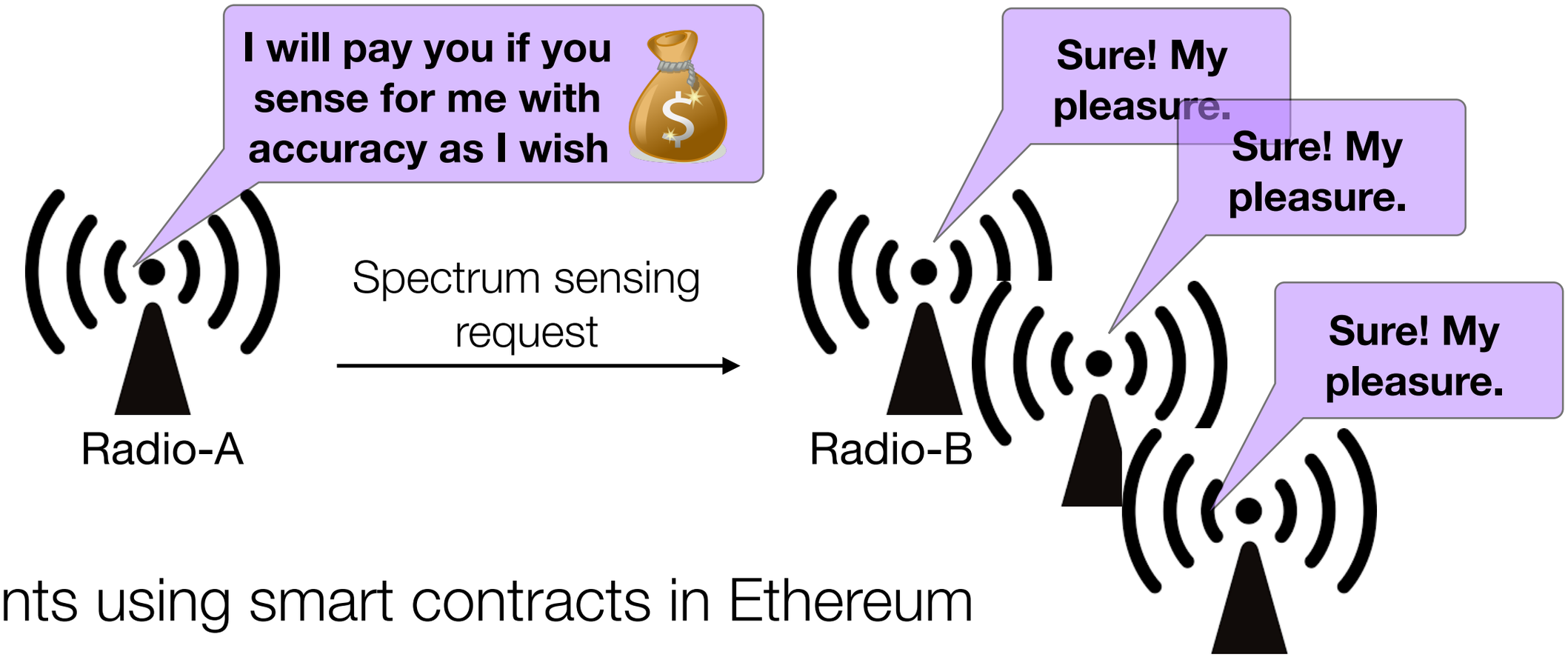
Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts



Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts

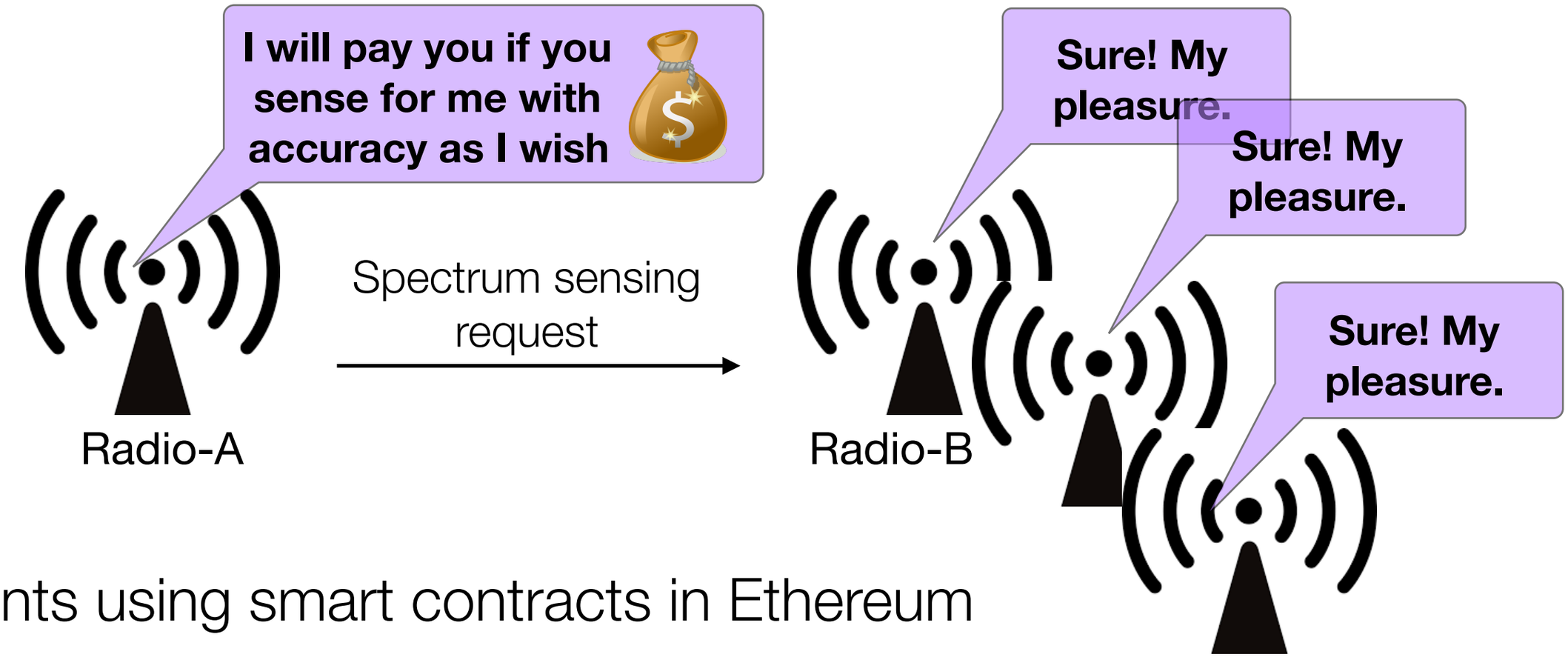


Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts



- payments using smart contracts in Ethereum
- but, smart contract usage in Ethereum is **not free**!

Spass: **Sp**ectrum Sensing **a**s a **S**ervice via **S**mart contracts



- payments using smart contracts in Ethereum
- but, smart contract usage in Ethereum is **not free**!

Can Spass provide a feasible business model?

Spass: System Model

- A SU mobile network operator (MNO) buys sensing service from the sensors (***helpers***) in its proximity
- SU MNO defines its requirement and payment for sensing service in a smart contract

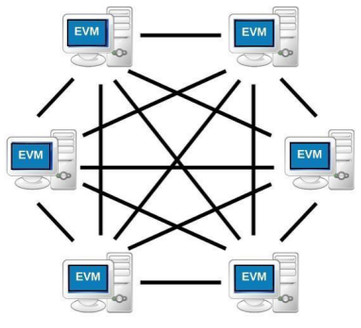
Spass: System Model

- A SU mobile network operator (MNO) buys sensing service from the sensors (***helpers***) in its proximity
- SU MNO defines its requirement and payment for sensing service in a smart contract

SU MNO has to pay:

Helpers

simple sensors to advanced spectrum analyzers



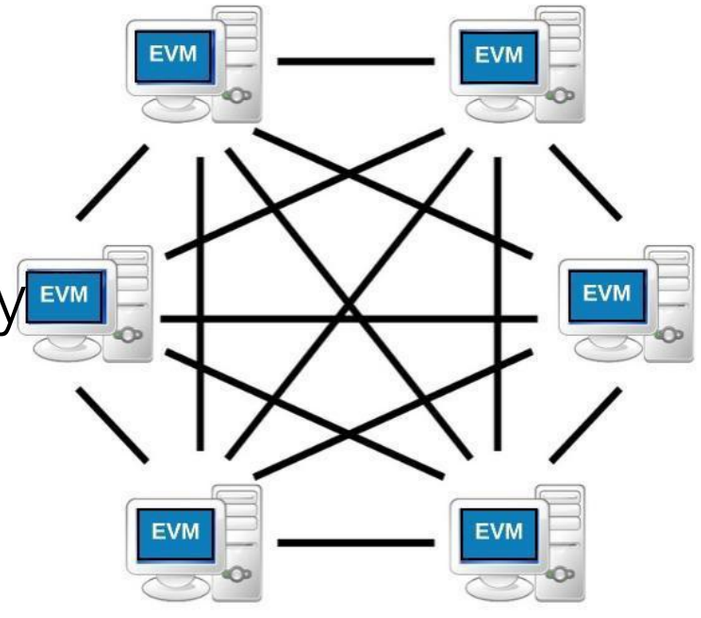
Ethereum smart contract network

Two key questions

1. What is the cost of using smart contract?
2. Under which conditions Spass can sustain a profitable business model?
    1. helpers willing to participate
    2. SU willing to buy service

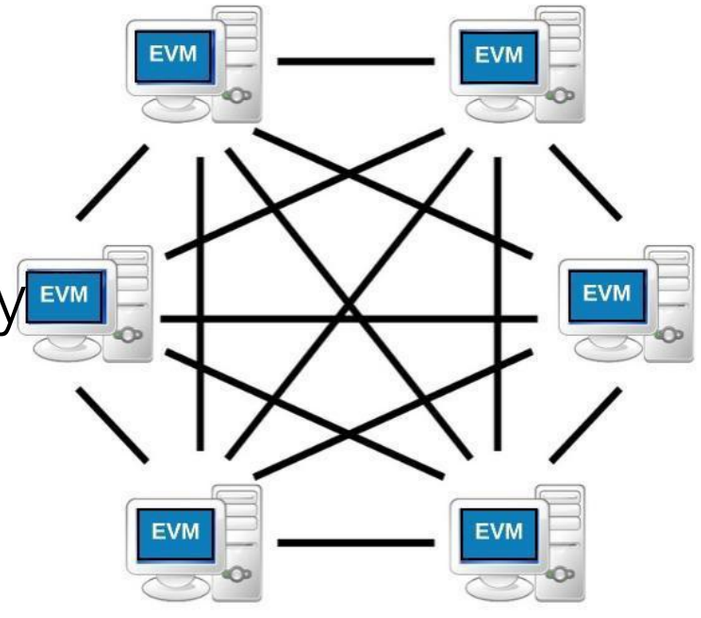
Overview of smart-contracts in Ethereum

- Computer program running on Blockchain, e.g., Ethereum
- Defining terms of an agreement
- No need for trust among trading entities, no intermediary
  - Trust is in the chain!
- **account** containing (immutable) code and storage
  - Can send Ether (credit) to other accounts
  - Read/write storage
  - Can call other contracts



Overview of smart-contracts in Ethereum

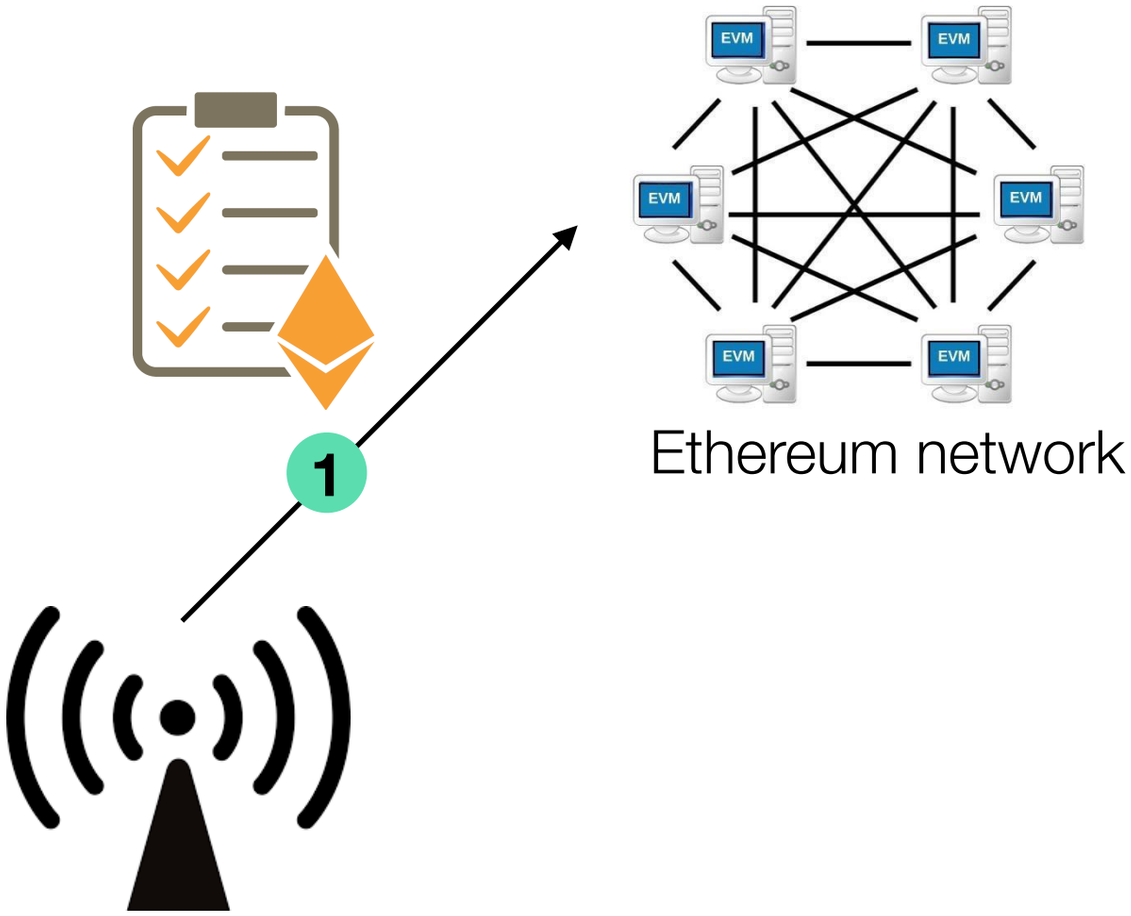
- Computer program running on Blockchain, e.g., Ethereum
- Defining terms of an agreement
- No need for trust among trading entities, no intermediary
  - Trust is in the chain!
- **account** containing (immutable) code and storage
  - Can send Ether (credit) to other accounts
  - Read/write storage
  - Can call other contracts



ETHEREUM

- Smart contracts charge the caller of a function! [To avoid DDoS attacks]
- The cost of a function is increasing with the amount of data written to the contract and complexity of computation.

Overview of Spass

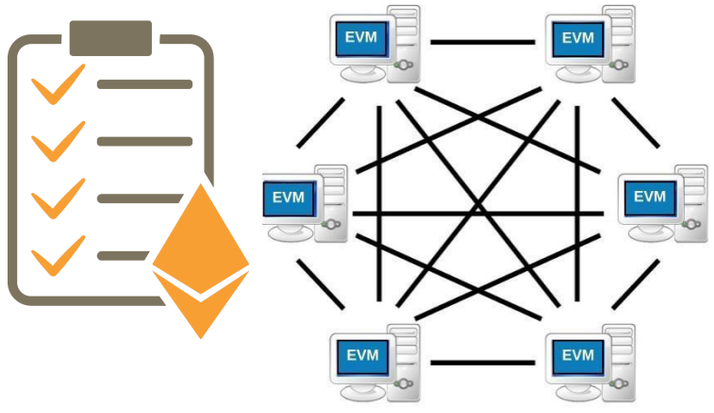


Ethereum network

1- Register Spass contract in Ethereum

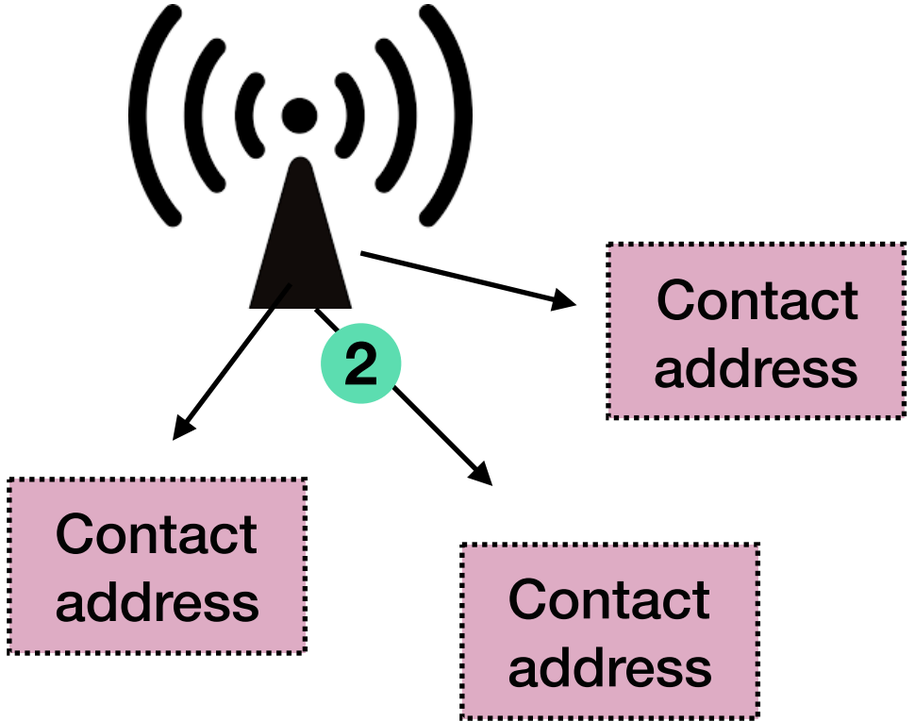
- SU defines the parameters
  - required minimum sensing accuracy in terms of PU detection accuracy, false alarm rate
  - payment
  - dispute resolution

Overview of Spass

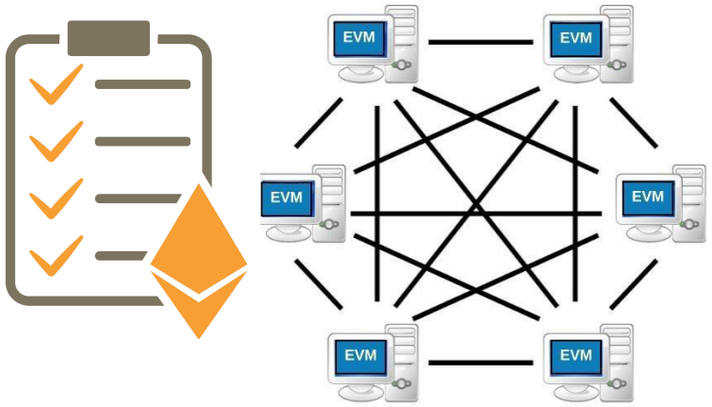


1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers



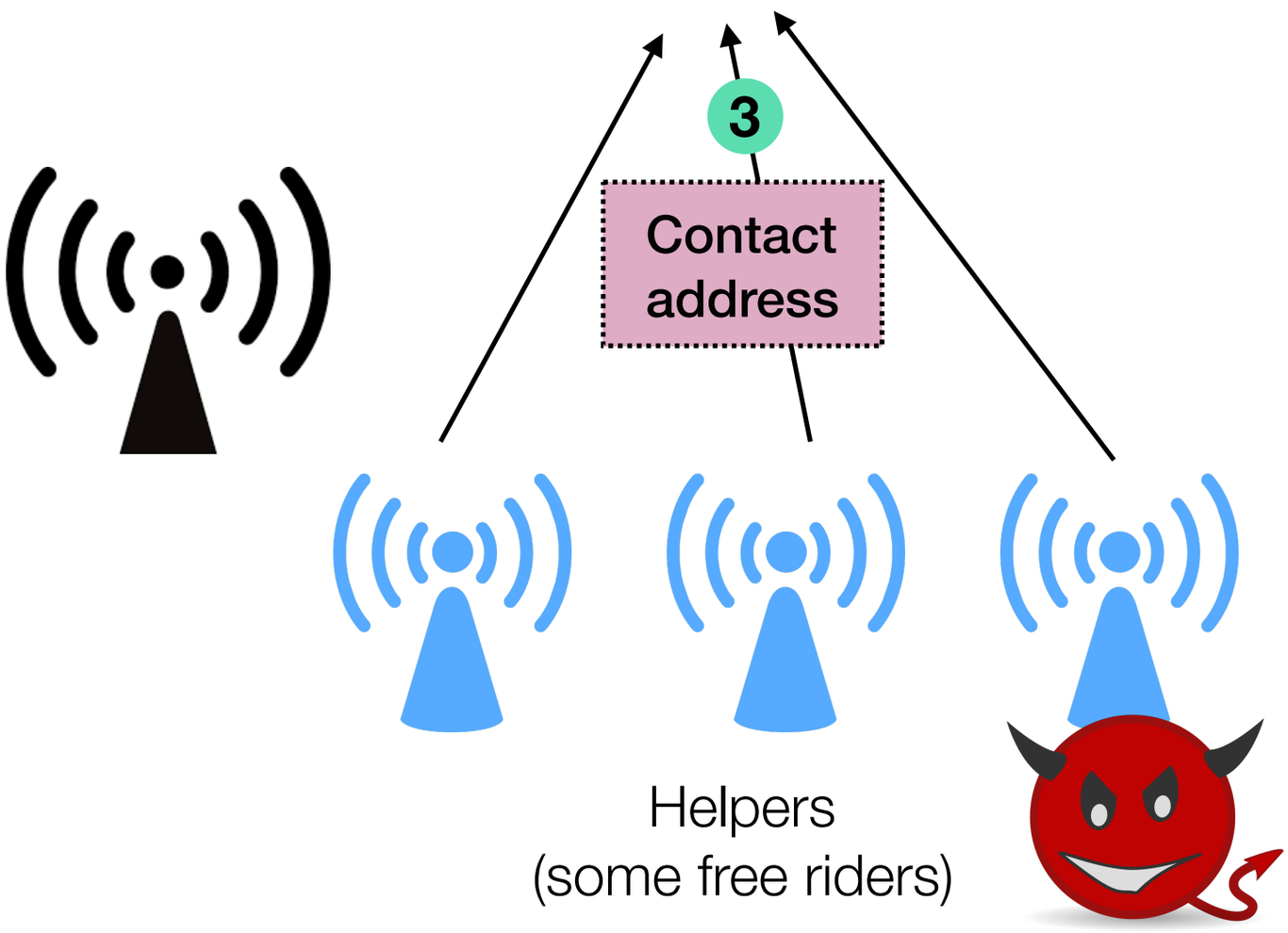
Overview of Spass



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

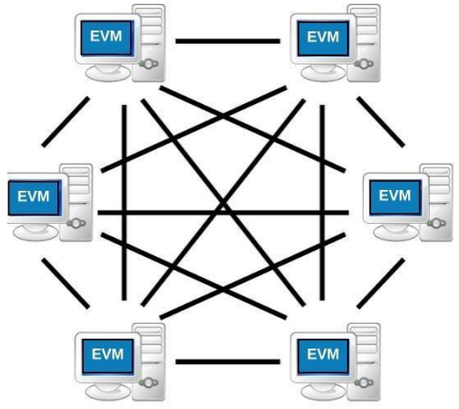
3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers



Overview of Spass

4

Helper selection



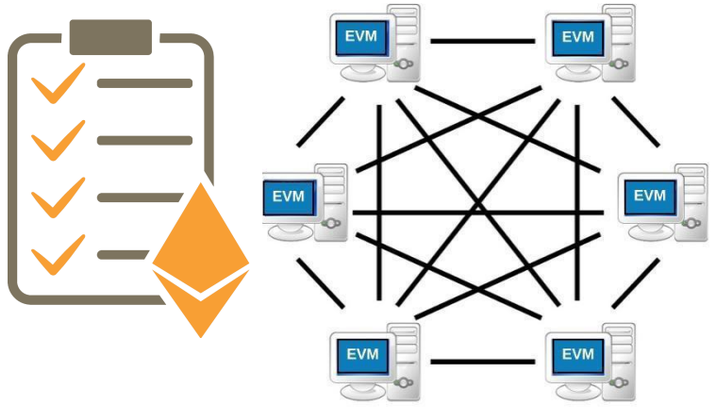
1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

Overview of Spass



1- Register Spass contract in Ethereum

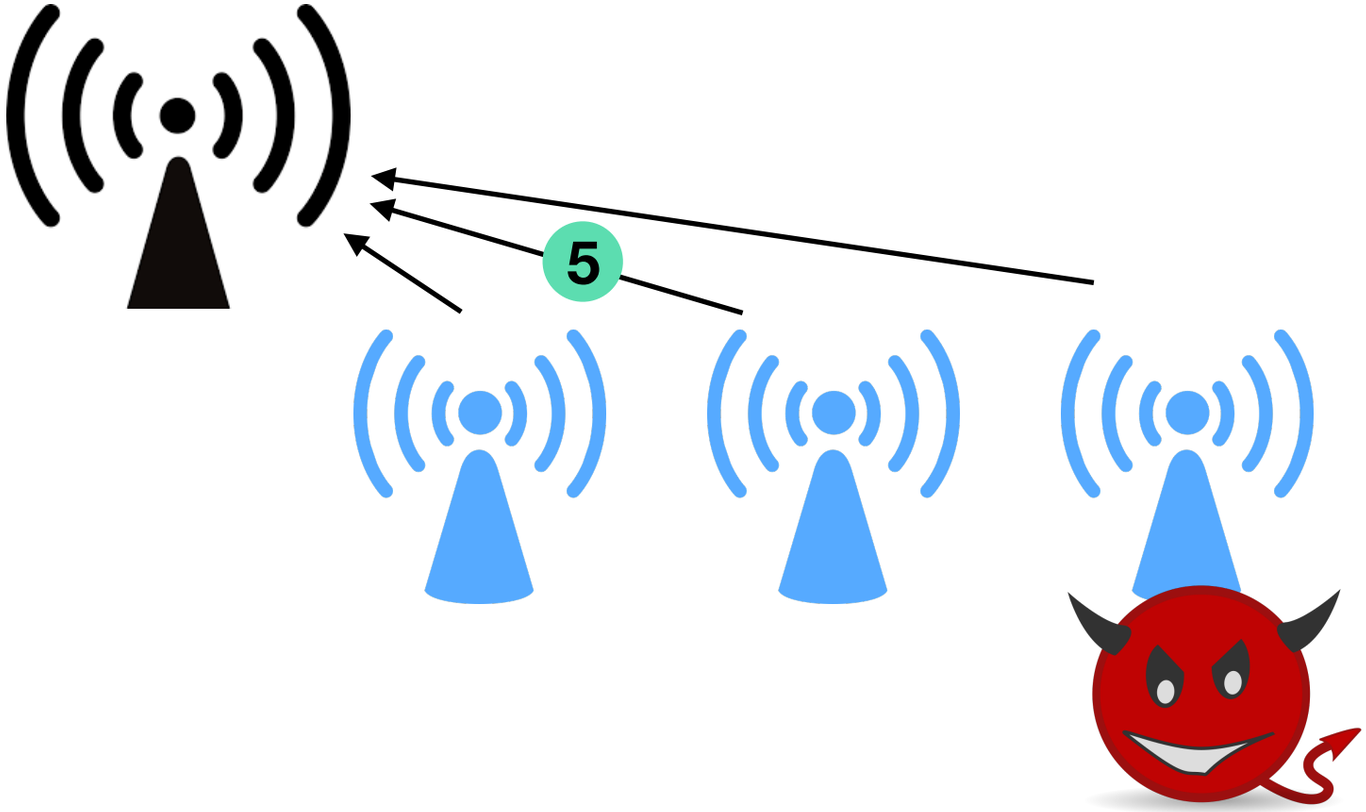
2- Broadcast the contract address to helpers

3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers

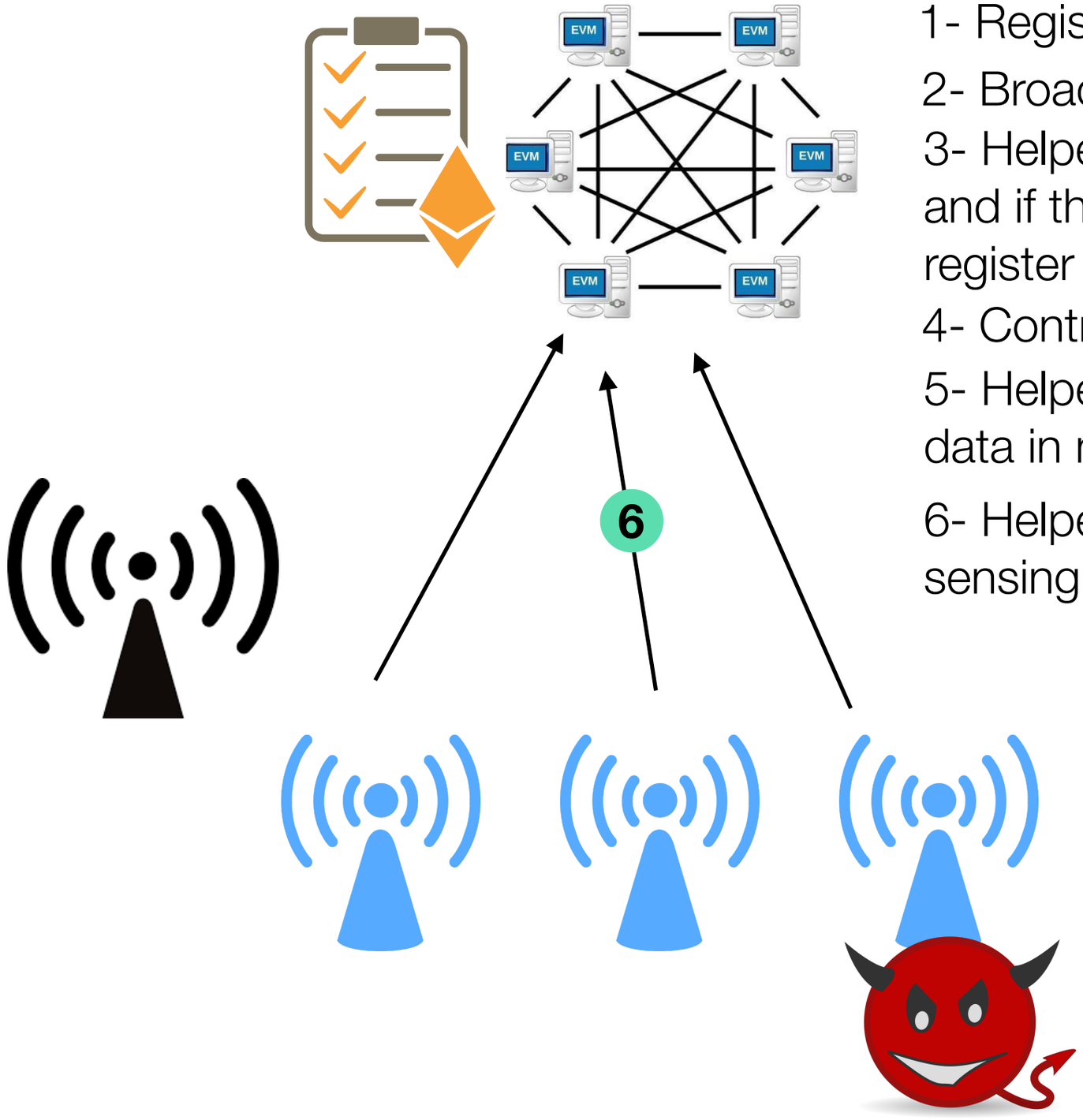
4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

Majority logic



Overview of Spass



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

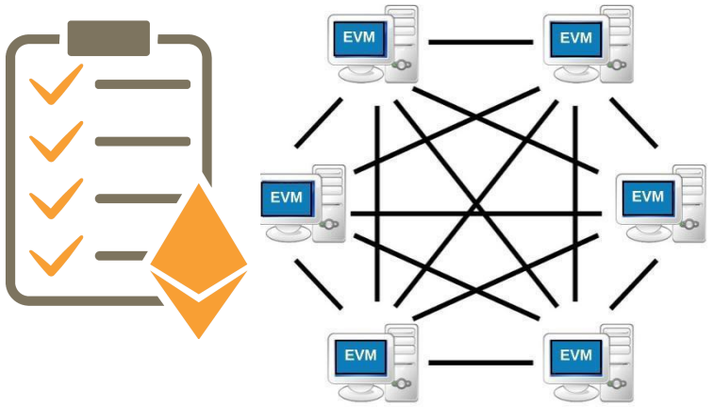
3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

6- Helper-Ethereum network **compressed** sensing reports periodically

Overview of Spass



Malicious helper identification

1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract and if the terms are suitable for them, they register as candidate helpers

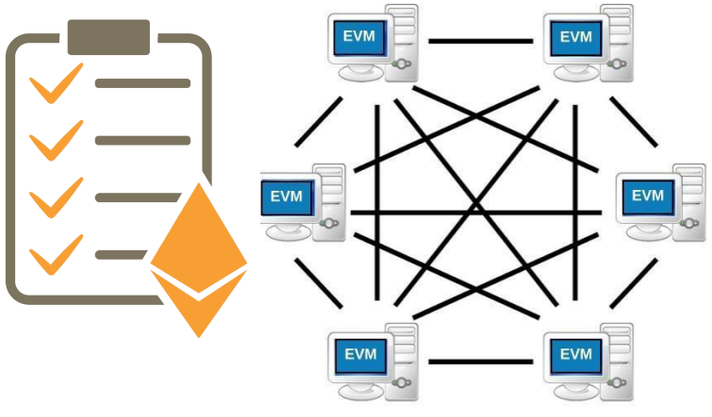
4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

Overview of Spass



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

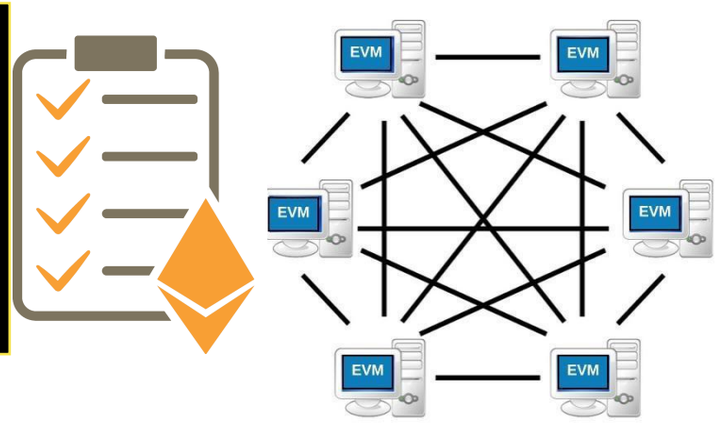
6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

Overview of Spass

Blacklisted helpers

[redacted]



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract
and if the terms are suitable for them, they register as candidate helpers

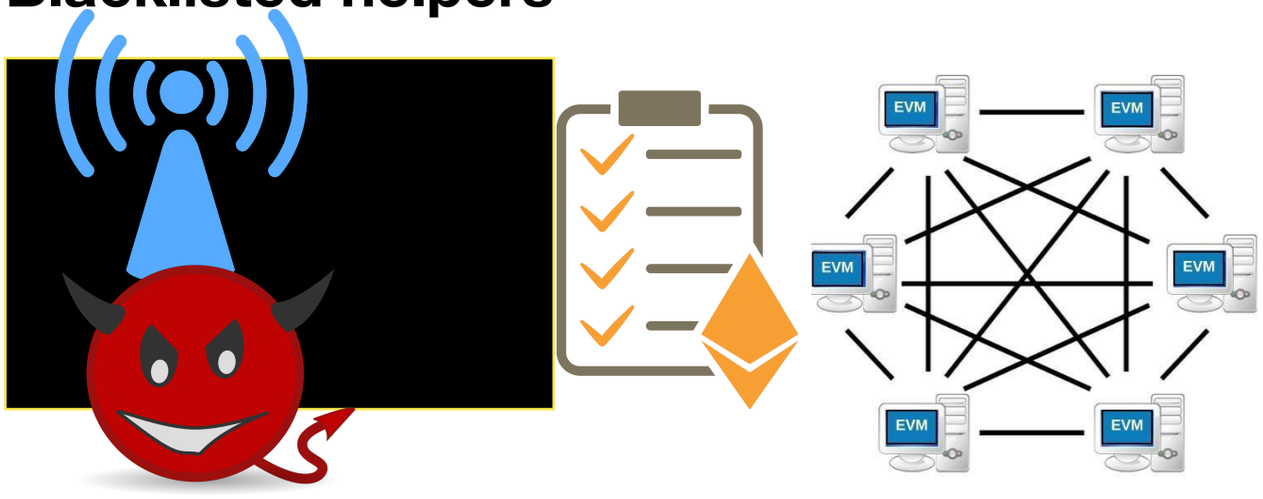
4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

Overview of Spass



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

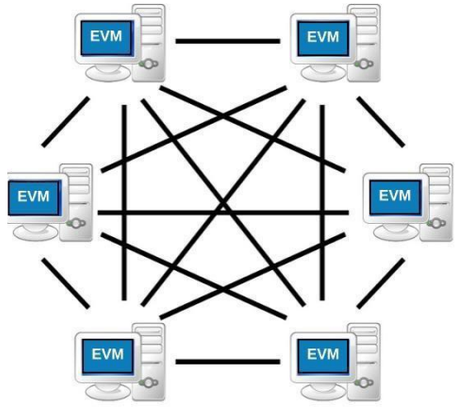
5- Helpers send to SU their actual sensing data in near-real time

6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

Overview of Spass

Blacklisted helpers



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

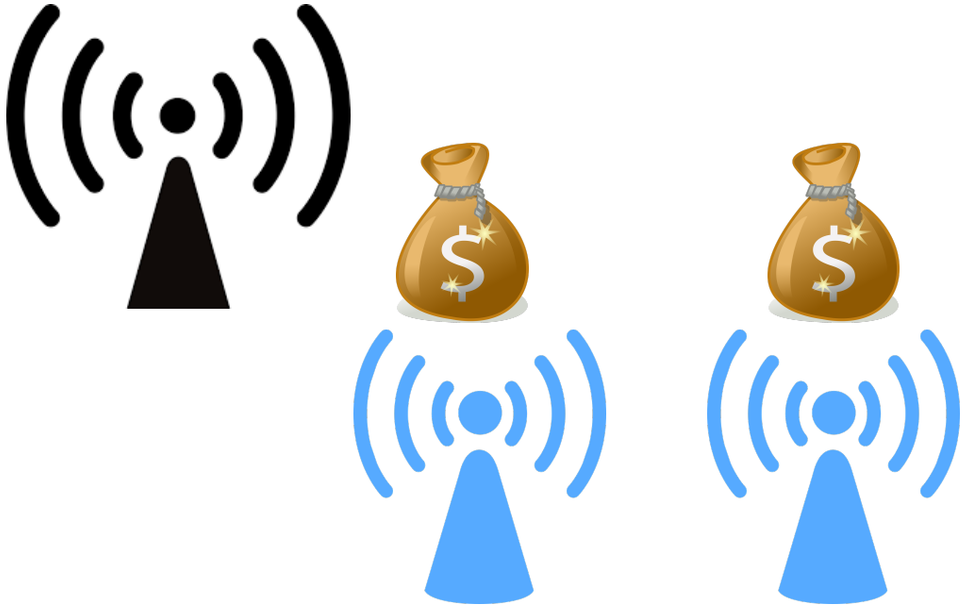
6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

8- Malicious helpers are **blacklisted**, the rest gets payment

Overview of Spass

Blacklisted helpers



1- Register Spass contract in Ethereum

2- Broadcast the contract address to helpers

3- Helpers check the contract and if the terms are suitable for them, they register as candidate helpers

4- Contract selects H helpers

5- Helpers send to SU their actual sensing data in near-real time

6- Helper-Ethereum network **compressed** sensing reports periodically

7- Contract runs **malicious helper identification** algorithm

8- Malicious helpers are **blacklisted**, the rest gets payment

Spass contract functionality

Helper selection 

Malicious helper identification 

Payment (clearing) 

Spass contract functionality

Helper selection 

Malicious helper identification 

Payment (clearing) 

- Based on price of sensing, sensing accuracy, reputations, …
- Goal:
  - satisfy spectrum sensing accuracy asserted by the regulator (high probability of detection)
  - discover available spectrum (low false alarm)

Spass contract functionality

Helper selection

Malicious helper identification

Payment (clearing)

- Based on collected sensing reports
  - Helpers report might arrive asynchronously and hence Spass needs to store them before running malicious helper identification algorithm on the reports
  - Ethereum storage is limited and costly

Spass contract functionality

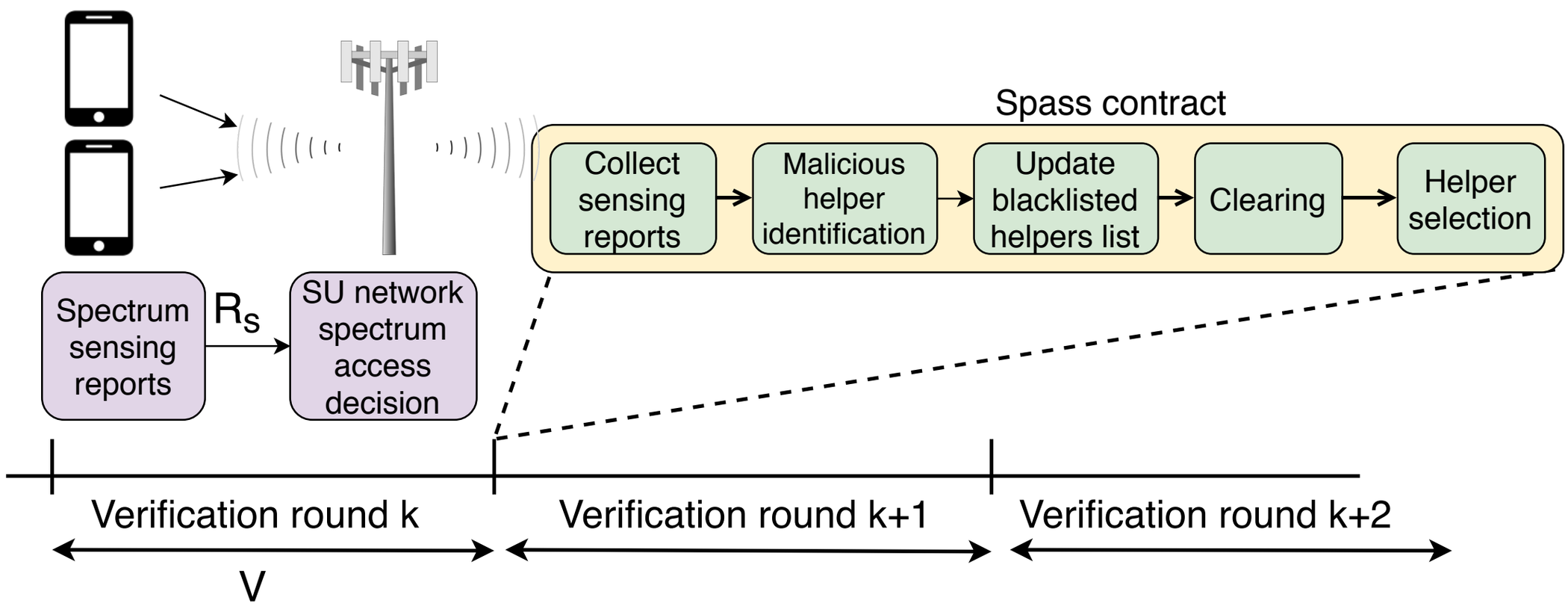
Helper selection

Malicious helper identification

Payment (clearing)

- Payment to helpers who are not blacklisted as malicious helpers

Verification rounds

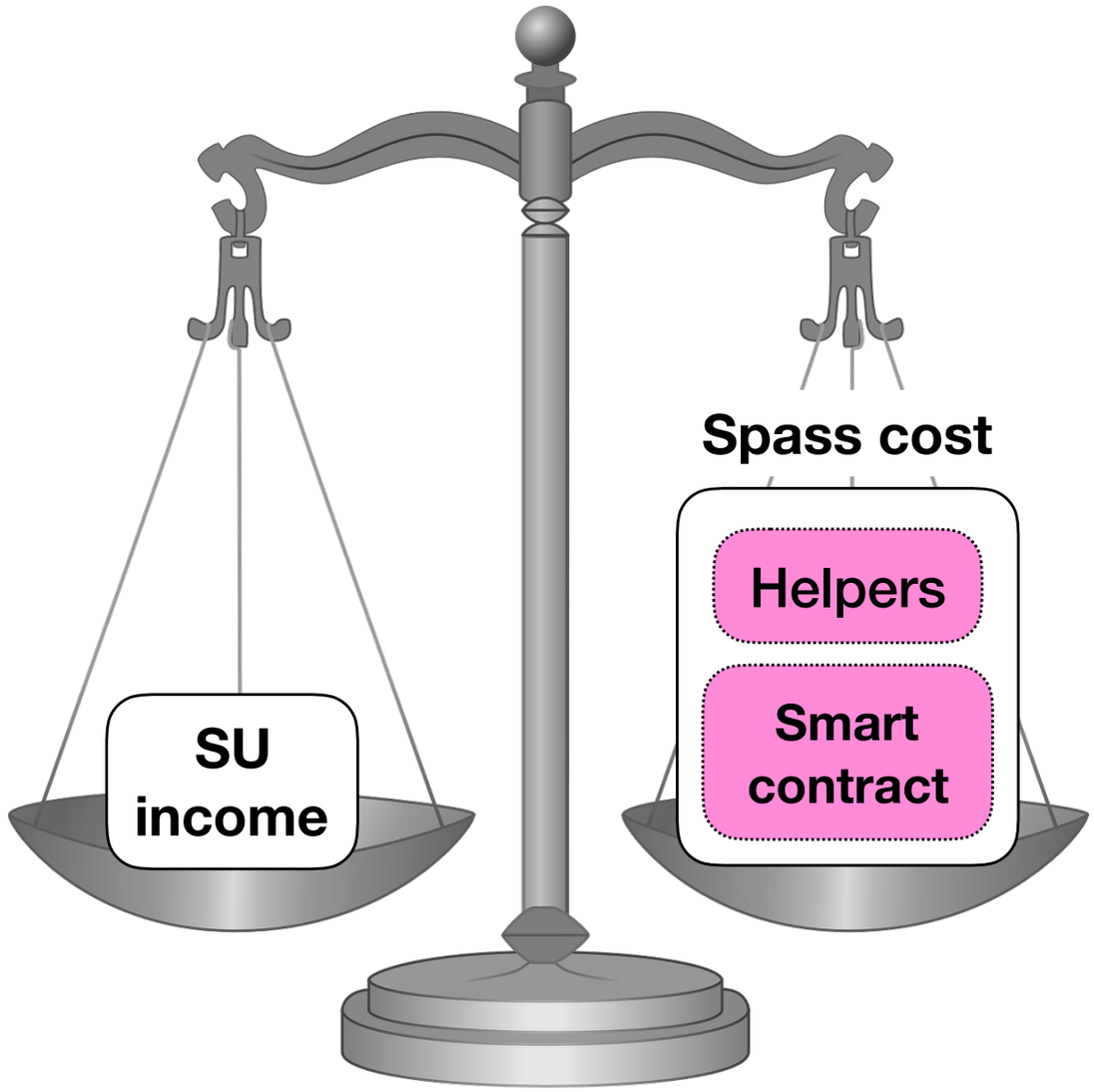


- Honest helpers (P_d^h, P_f^h)
- Malicious helpers (P_d^m, P_f^m): does not sense the spectrum but generates sensing data according to **PU's statistic p_0**
- Fraction of malicious helpers known to the contract

Business feasibility analysis

Income: How much money an SU network can earn by the discovered spectrum?

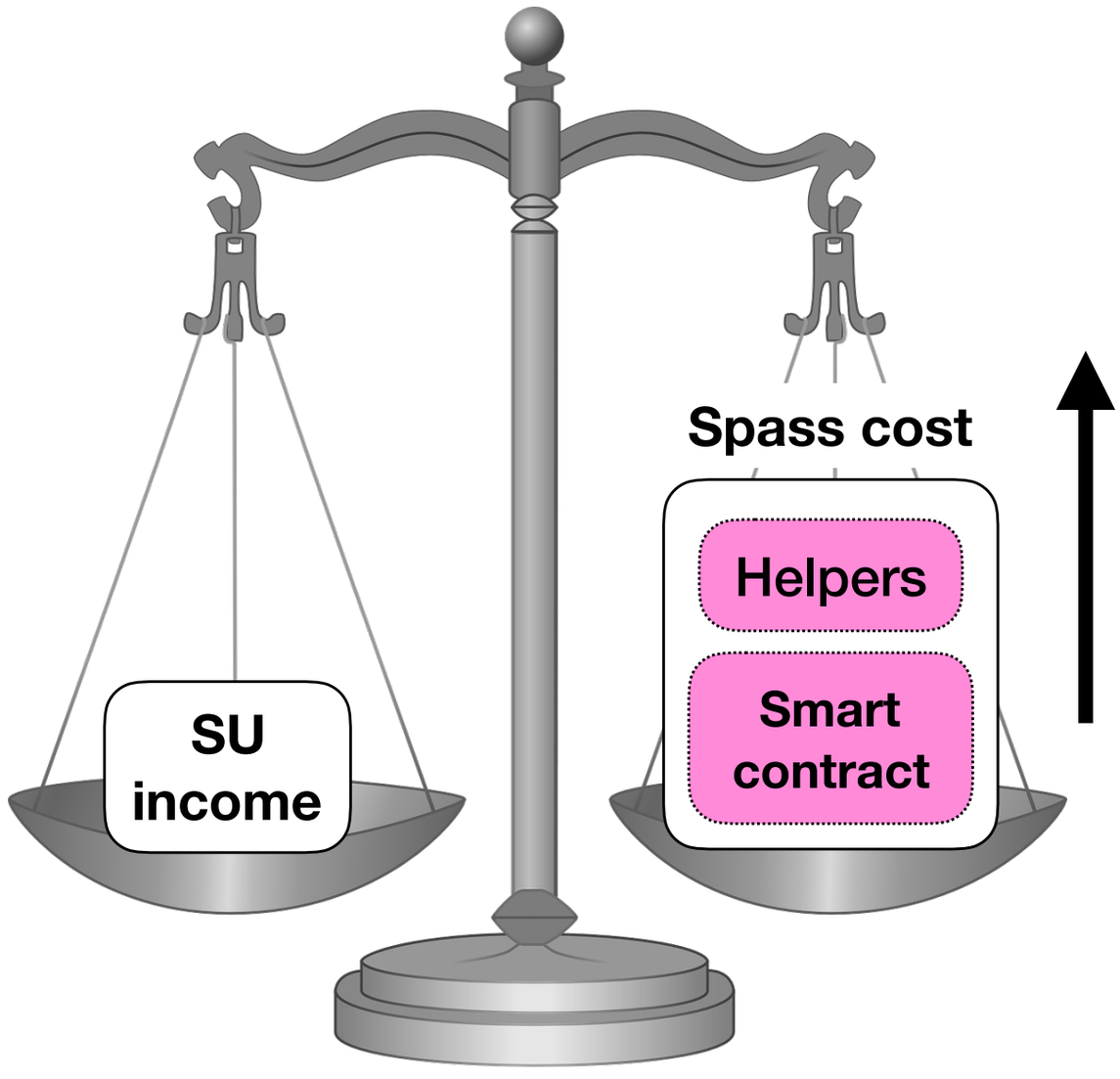
Payment: How much does it have to pay for Spass?



Business feasibility analysis

Income: How much money an SU network can earn by the discovered spectrum?

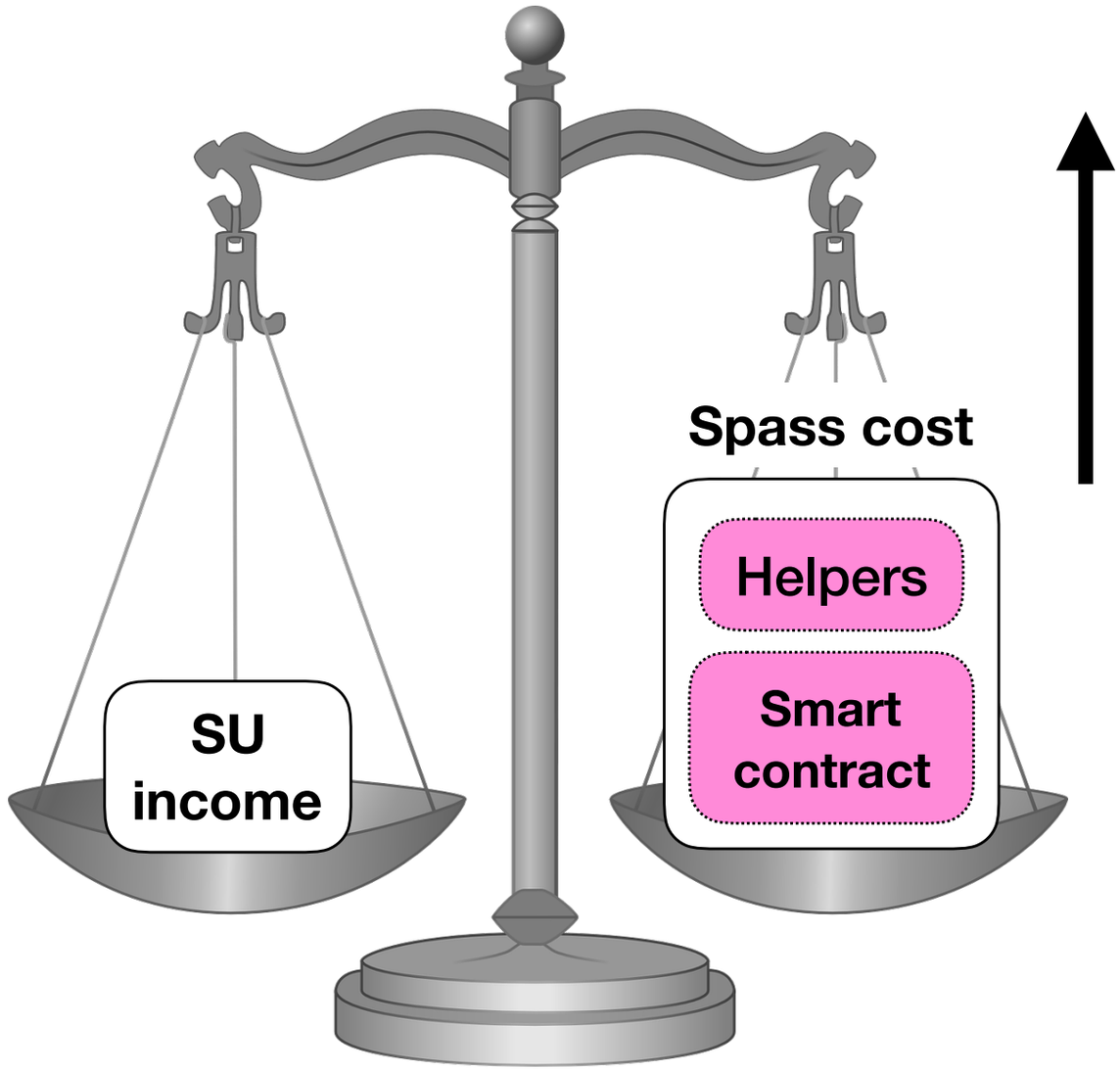
Payment: How much does it have to pay for Spass?



Business feasibility analysis

Income: How much money an SU network can earn by the discovered spectrum?

Payment: How much does it have to pay for Spass?



Income of the SU network

$$\Upsilon^{+} = \mu \times \kappa \times \mathcal{U}^{\text{Spass}} \times B \quad \text{€ per second.}$$

- A client pays μ euros/bit for second
- Channel bandwidth B
- U^{spass}: Utility of the discovered spectrum: $(1-p_f)p_0$
- Spectral efficiency of the SU network: κ bits/sec/Hertz

No false alarm

PU is idle

Payment to Spass

- We assume **write operation** to the contract is the dominant cost

$$\Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H \quad € \text{ per second.}$$

number of helpers

Payment to Spass

- We assume **write operation** to the contract is the dominant cost

$$\Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H \quad € \text{ per second.}$$

number of helpers

rate of sensing

Payment to Spass

- We assume **write operation** to the contract is the dominant cost

$$\Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H \quad € \text{ per second.}$$

number of helpers

cost of sensing

rate of sensing

Payment to Spass

- We assume **write operation** to the contract is the dominant cost

$$\Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H \quad € \text{ per second.}$$

cost of Ethereum

cost of sensing

rate of sensing

number of helpers

Payment to Spass

- We assume **write operation** to the contract is the dominant cost

$$\Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H \quad € \text{ per second.}$$

cost of Ethereum

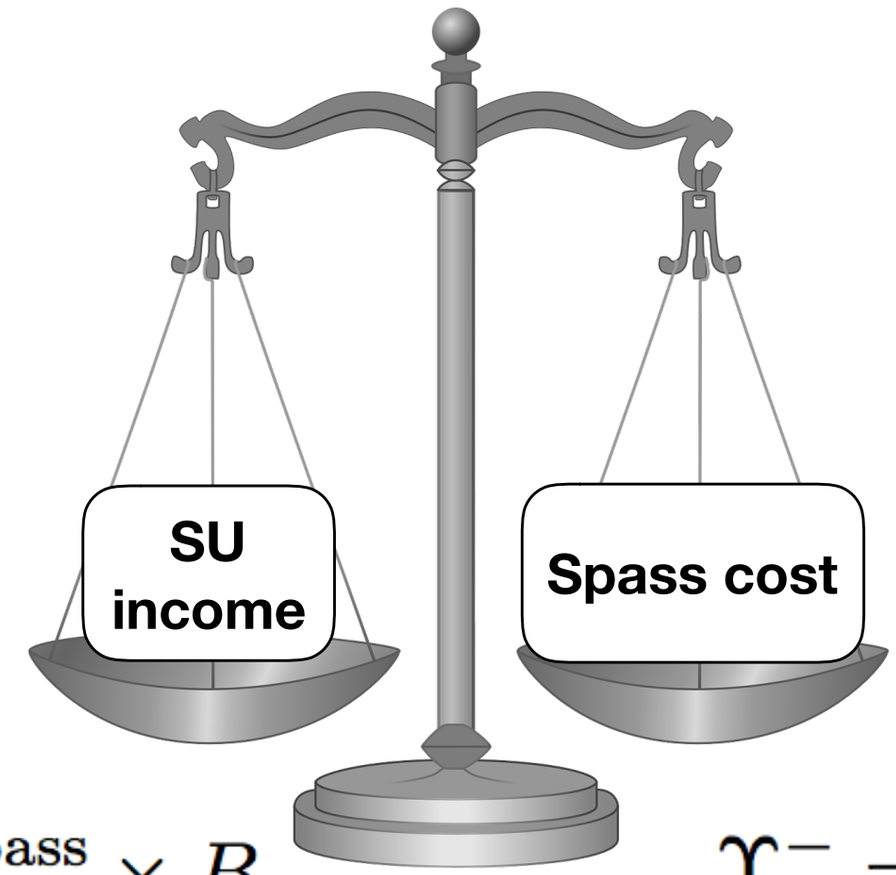
Compression ratio

cost of sensing

rate of sensing

number of helpers

Business feasibility analysis



$$\Upsilon^{+} = \mu \times \kappa \times \mathcal{U}^{\mathrm{Spass}} \times B \qquad \Upsilon^{-} = (\mu_{eth}\frac{R_s}{\beta} + \mu_s R_s) \times H$$

We find the operation range of an SU network in which it can make profit from Spass: $\Delta\Upsilon = \Upsilon^{+} - \Upsilon^{-} > 0$

When Spass is profitable?

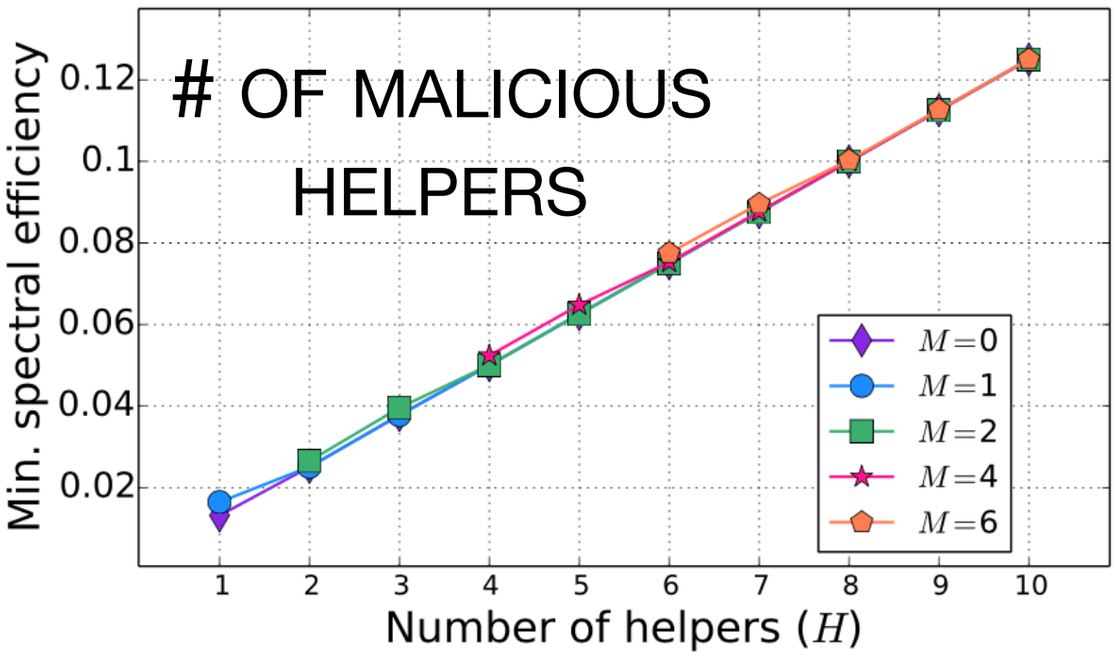
Required spectral efficiency

$$\kappa > \frac{R_s(\mu_{eth}/\beta + \mu_s) \times H}{\mu \times \mathcal{U}^{\text{Spass}} \times B}$$

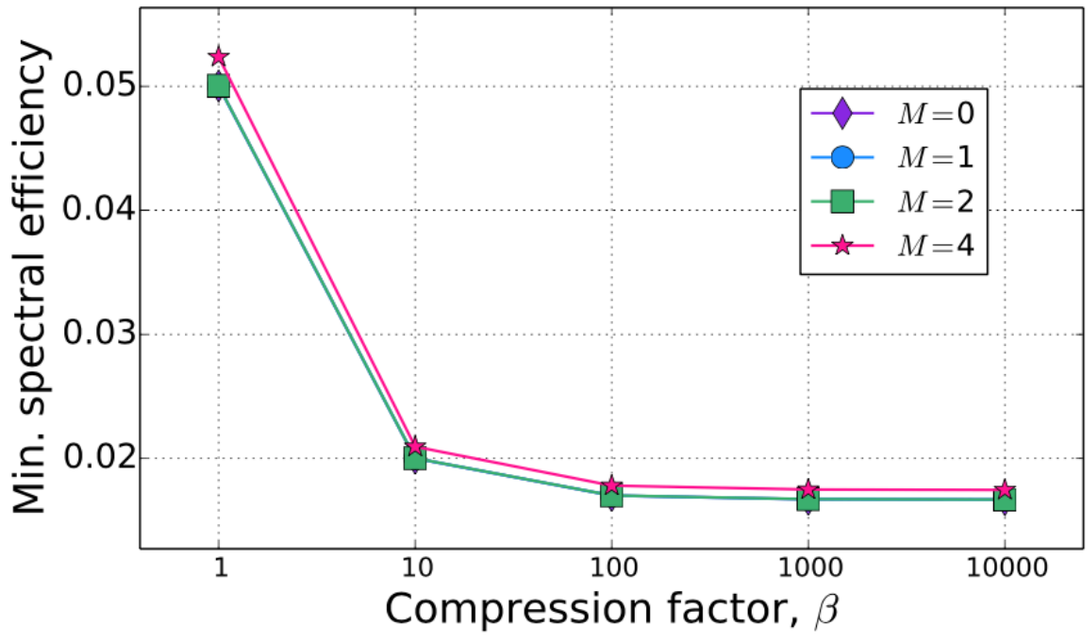
When Spass is profitable?

Required spectral efficiency

$$\kappa > \frac{R_s(\mu_{eth}/\beta + \mu_s) \times H}{\mu \times \mathcal{U}^{\text{Spass}} \times B}$$



(a) Impact of number of helpers H under $\beta = 1$.

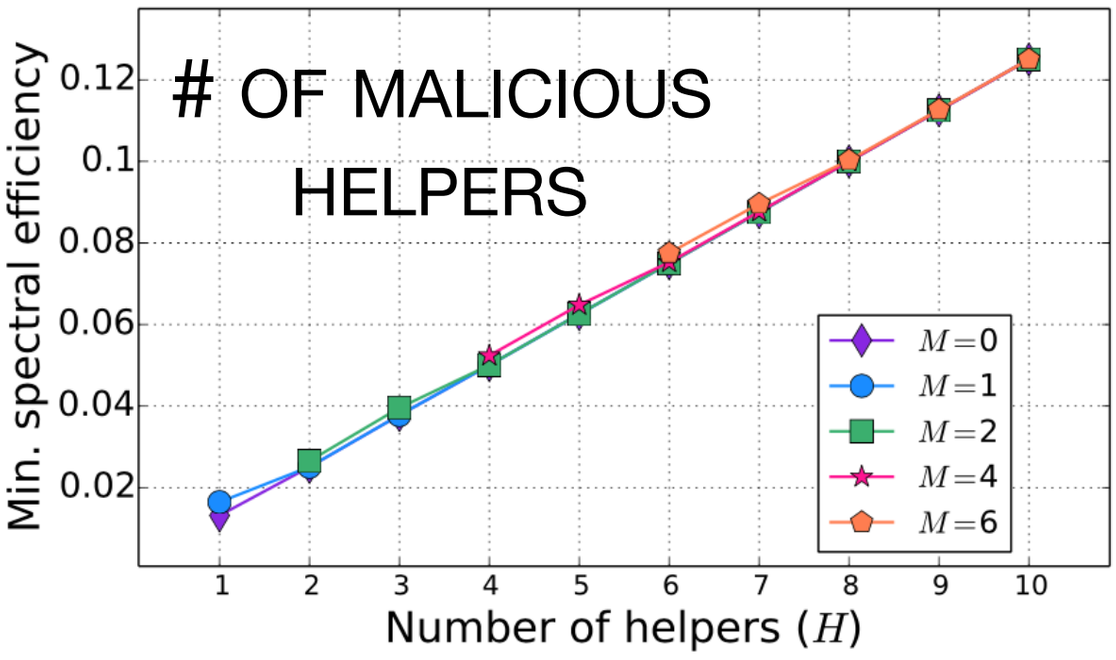


(b) Impact of compression β under $H = 4$.

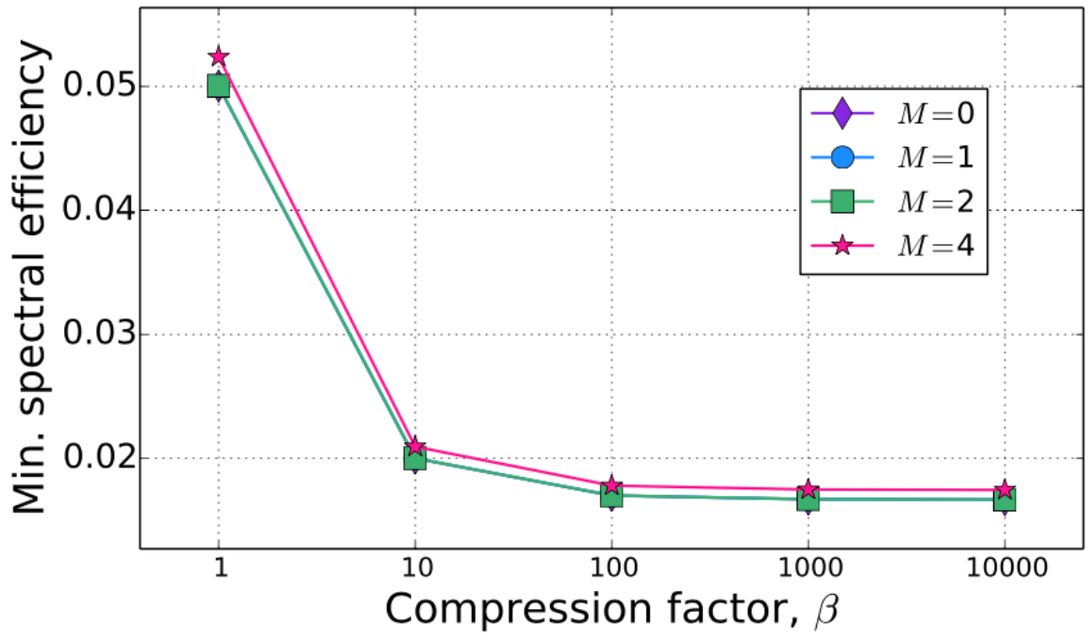
When Spass is profitable?

Required spectral efficiency

$$\kappa > \frac{R_s(\mu_{eth}/\beta + \mu_s) \times H}{\mu \times \mathcal{U}^{\text{Spass}} \times B}$$



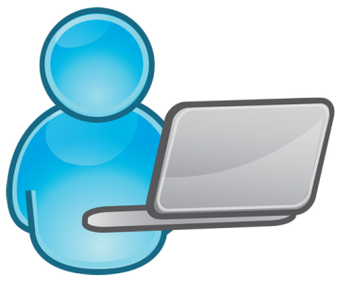
(a) Impact of number of helpers H under $\beta = 1$.



(b) Impact of compression β under $H = 4$.

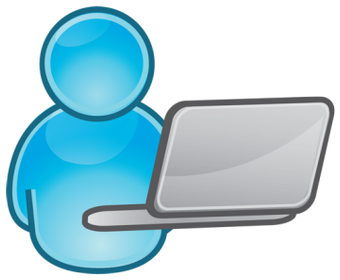
- Spass should identify malicious helpers
- Spass should implement high compression of reports

Helpers' reporting process



Sensing report: 10110010011100

Helpers' reporting process

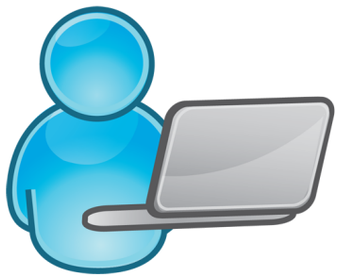


Sensing report: **1**011**1**00**1**00**1**10**0**

Compression factor: 1/3

Compressed sensing report: **11110**

Helpers' reporting process



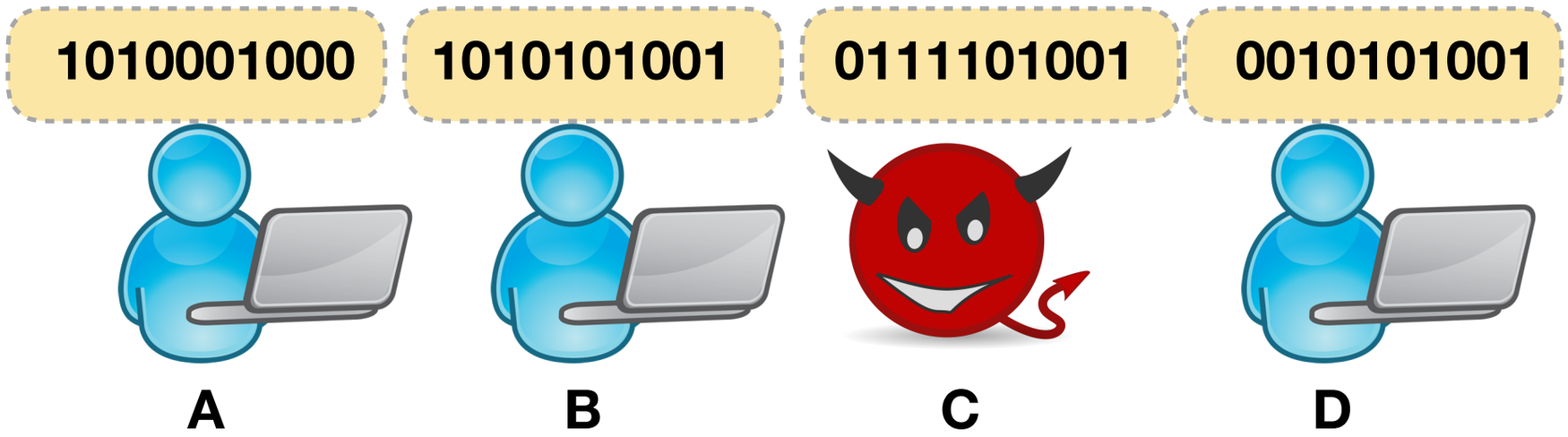
Sensing report: **1**011**1**00**1**00**1**10**0**

Compression factor: 1/3

Compressed sensing report: **11110**

Lower number of bits: lower cost but might result in lower malicious helper detection accuracy

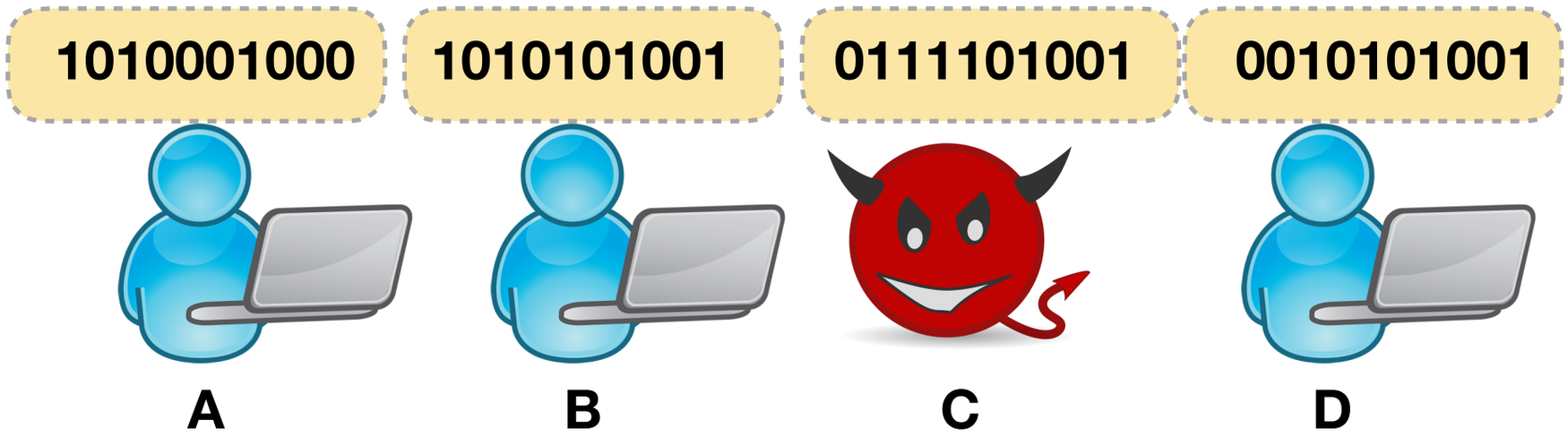
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

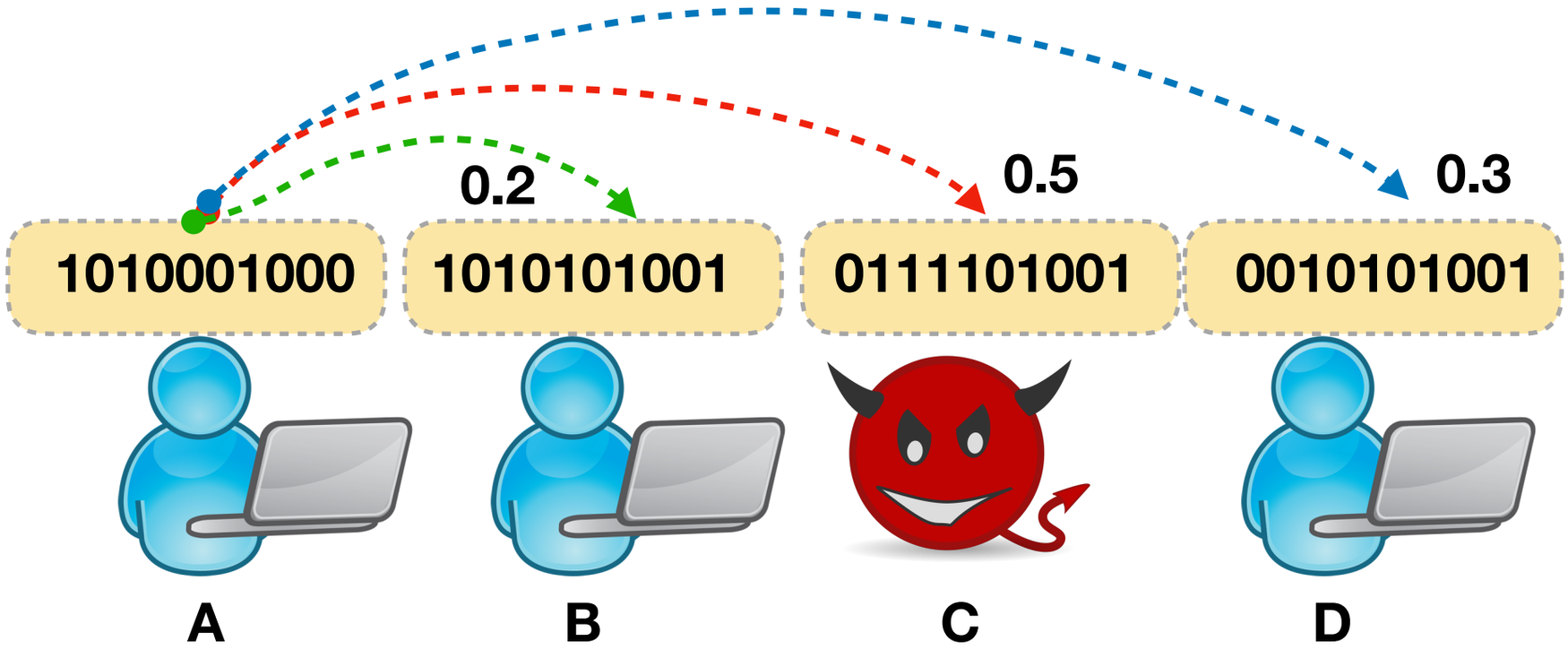
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

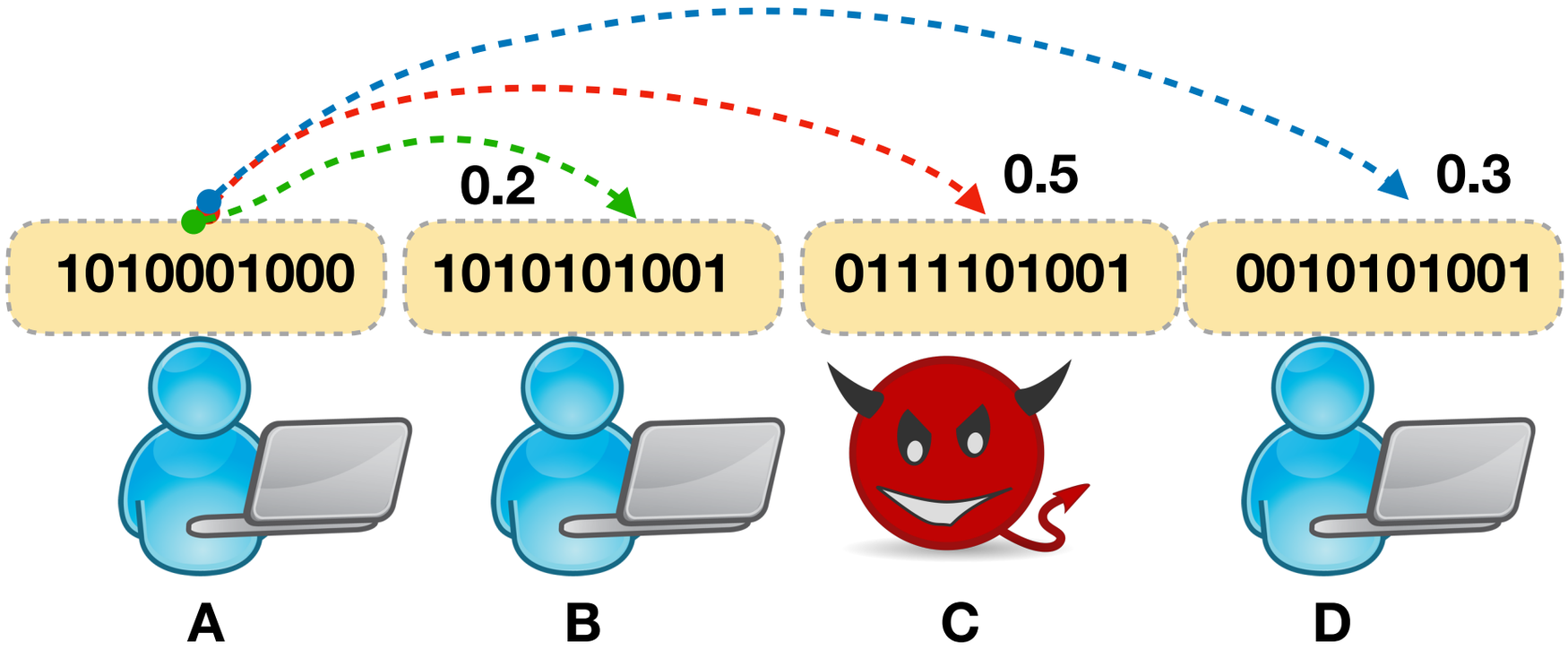
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

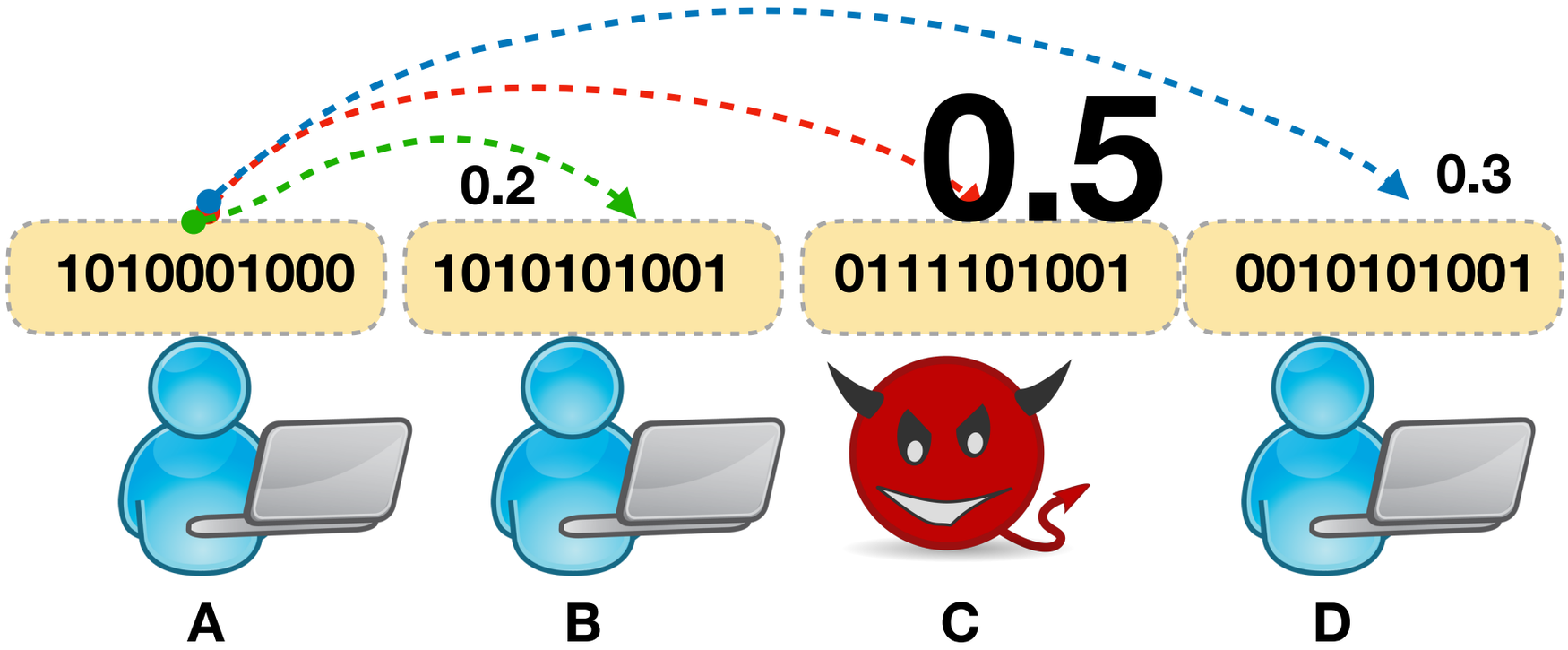
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

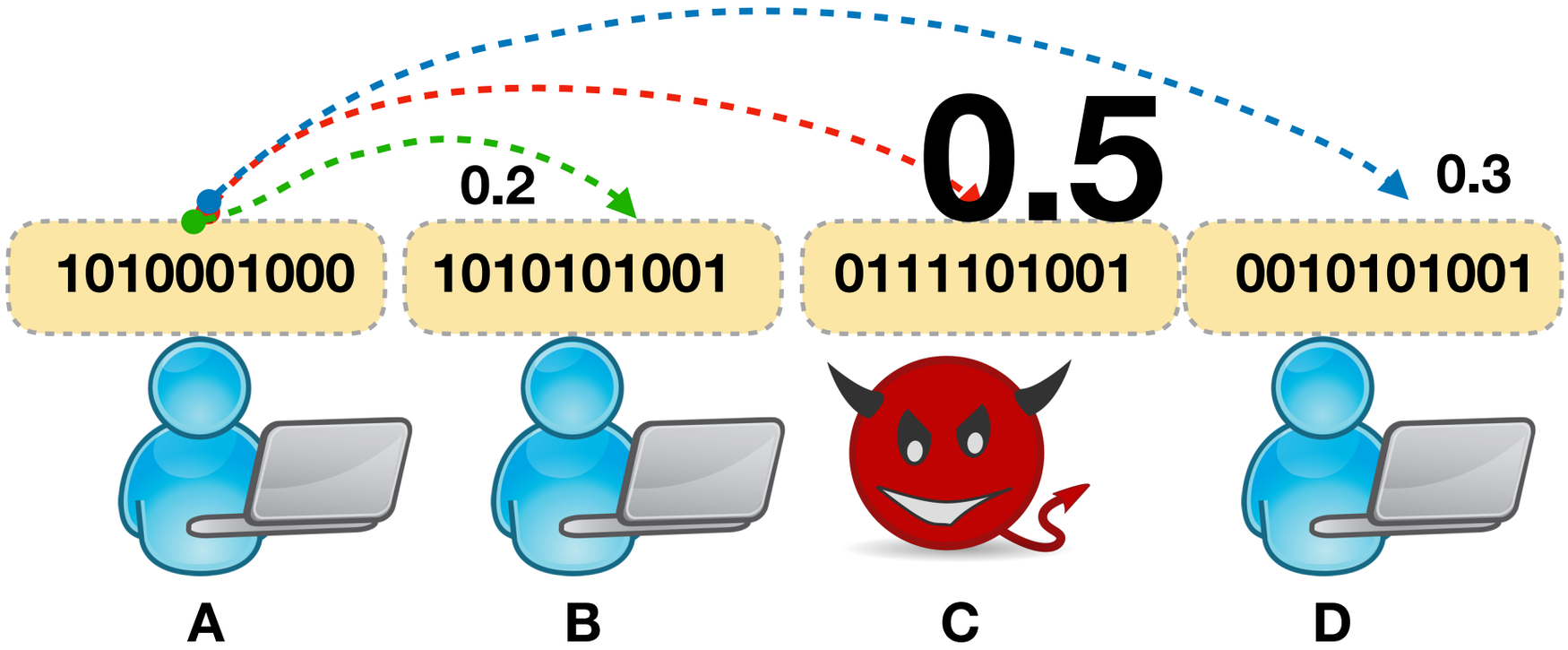
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

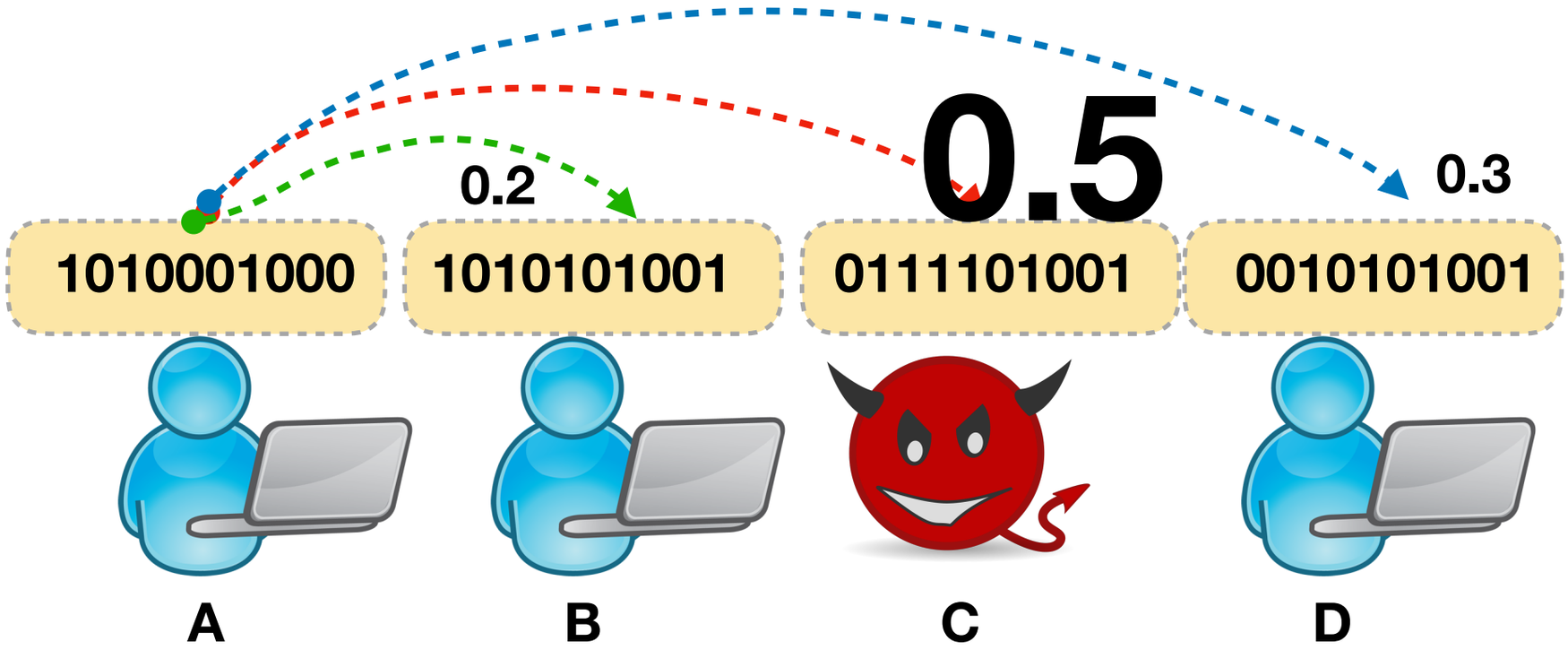
Two-threshold based voting



- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

Two-threshold based voting



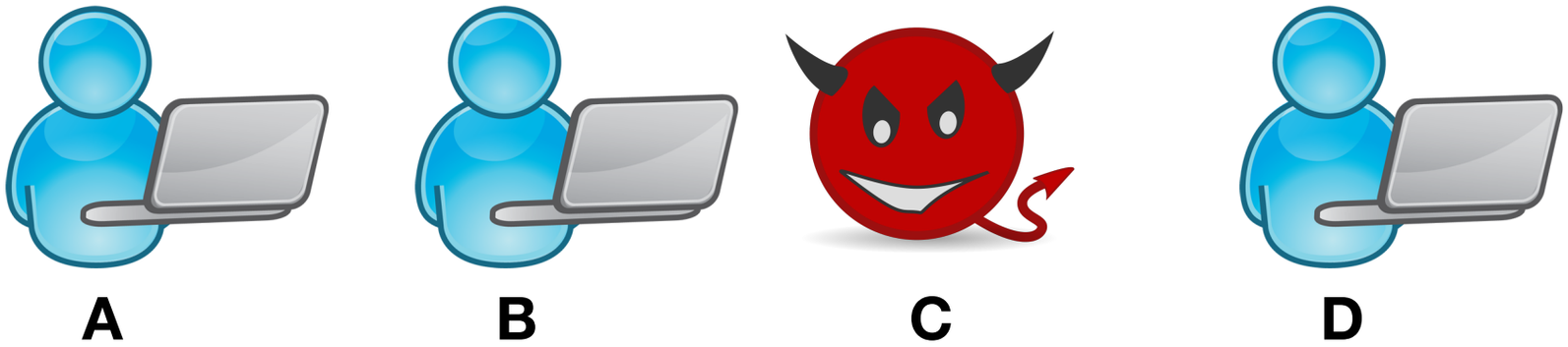
- **Distance between two helpers**
  - normalized Hamming distance of two helper reports [Li2010]
- **Expected distance**
  - honest-honest helpers
  - honest-malicious helpers

H. Li and Z. Han, “Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in CRNs,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3554–3565, 2010

Two-threshold based voting (TTBV) for each helper

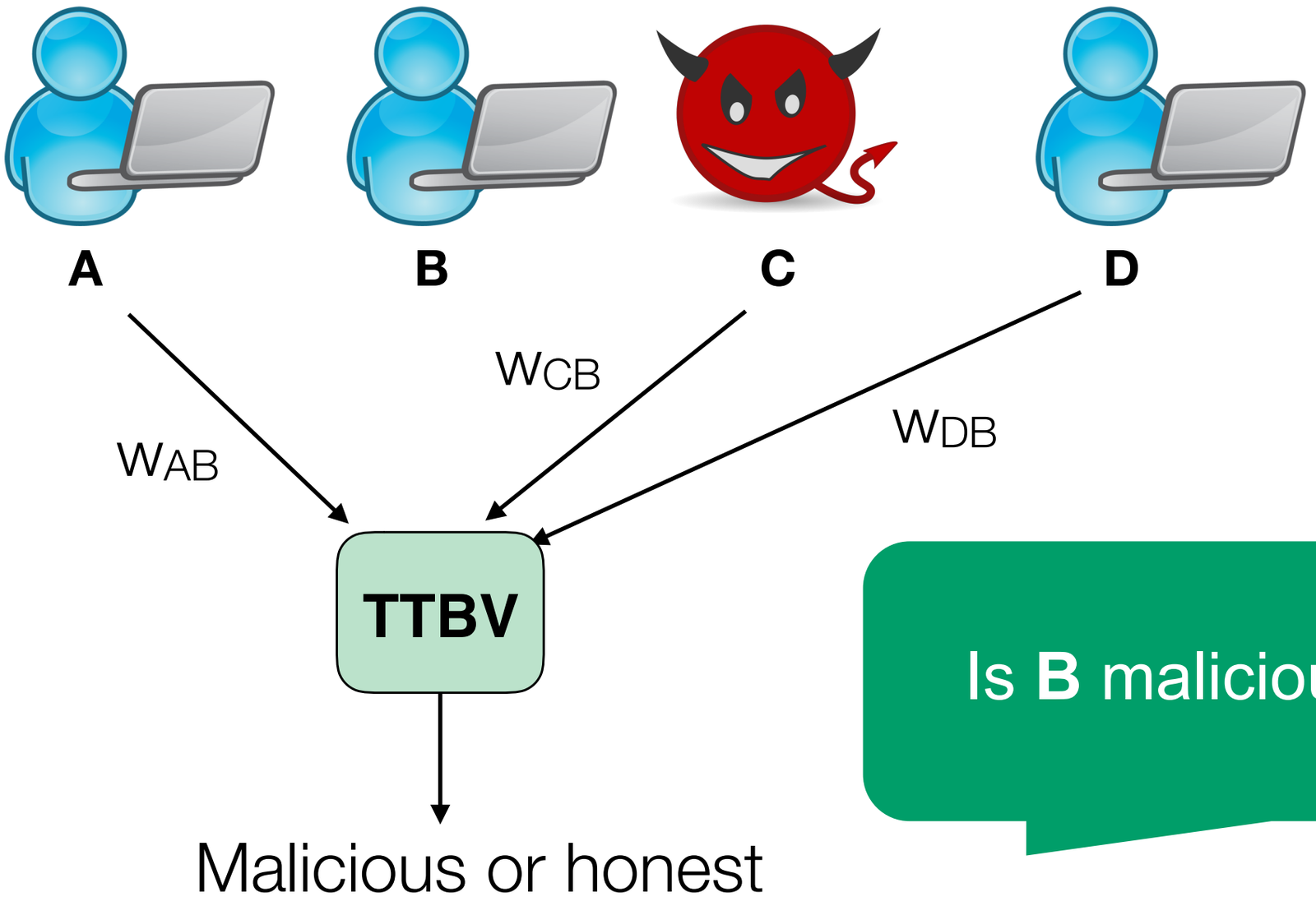
Two-threshold based voting (TTBV) for each helper

Two-threshold based voting (TTBV) for each helper



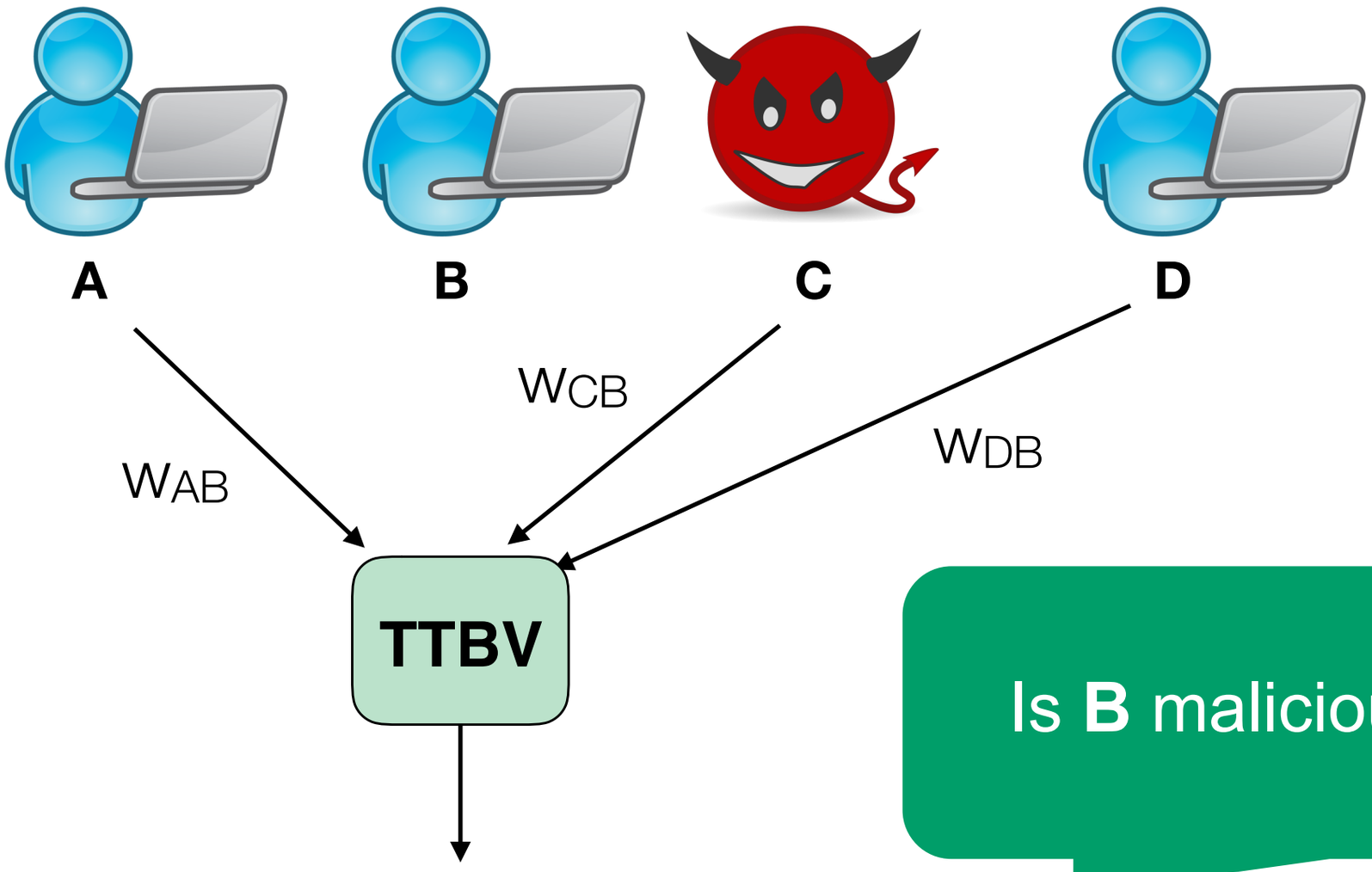
Is **B** malicious or honest?

Two-threshold based voting (TTBV) for each helper



Is **B** malicious or honest?

Two-threshold based voting (TTBV) for each helper

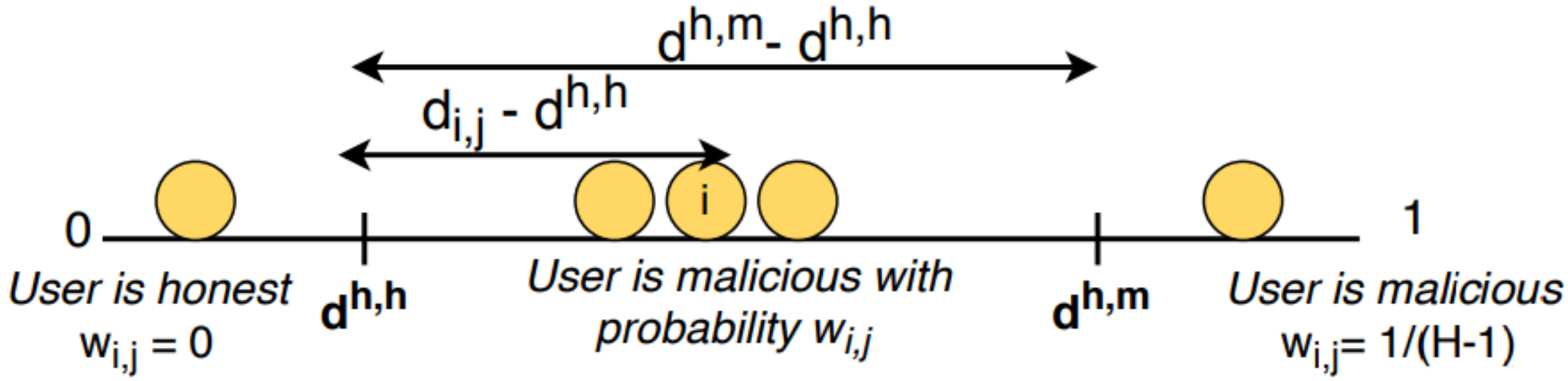


Is **B** malicious or honest?

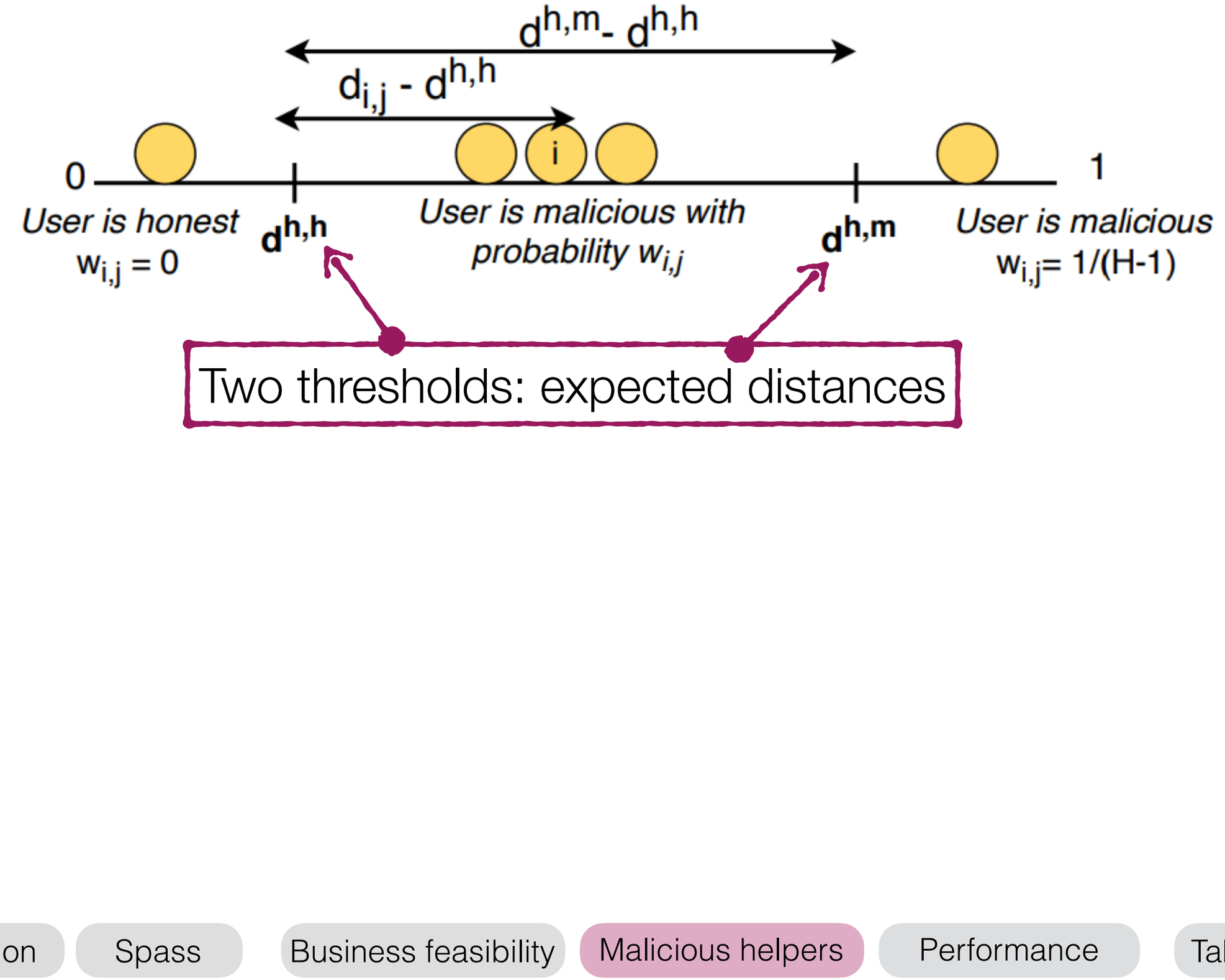
$$\text{State of } h_j = \begin{cases} \text{Malicious} & \text{if } \sum_{i,i\neq j} w_{i,j} \geqslant w_{th} \\ \text{Honest} & \text{else.} \end{cases}$$

w_{th} is a threshold greater than 0.5

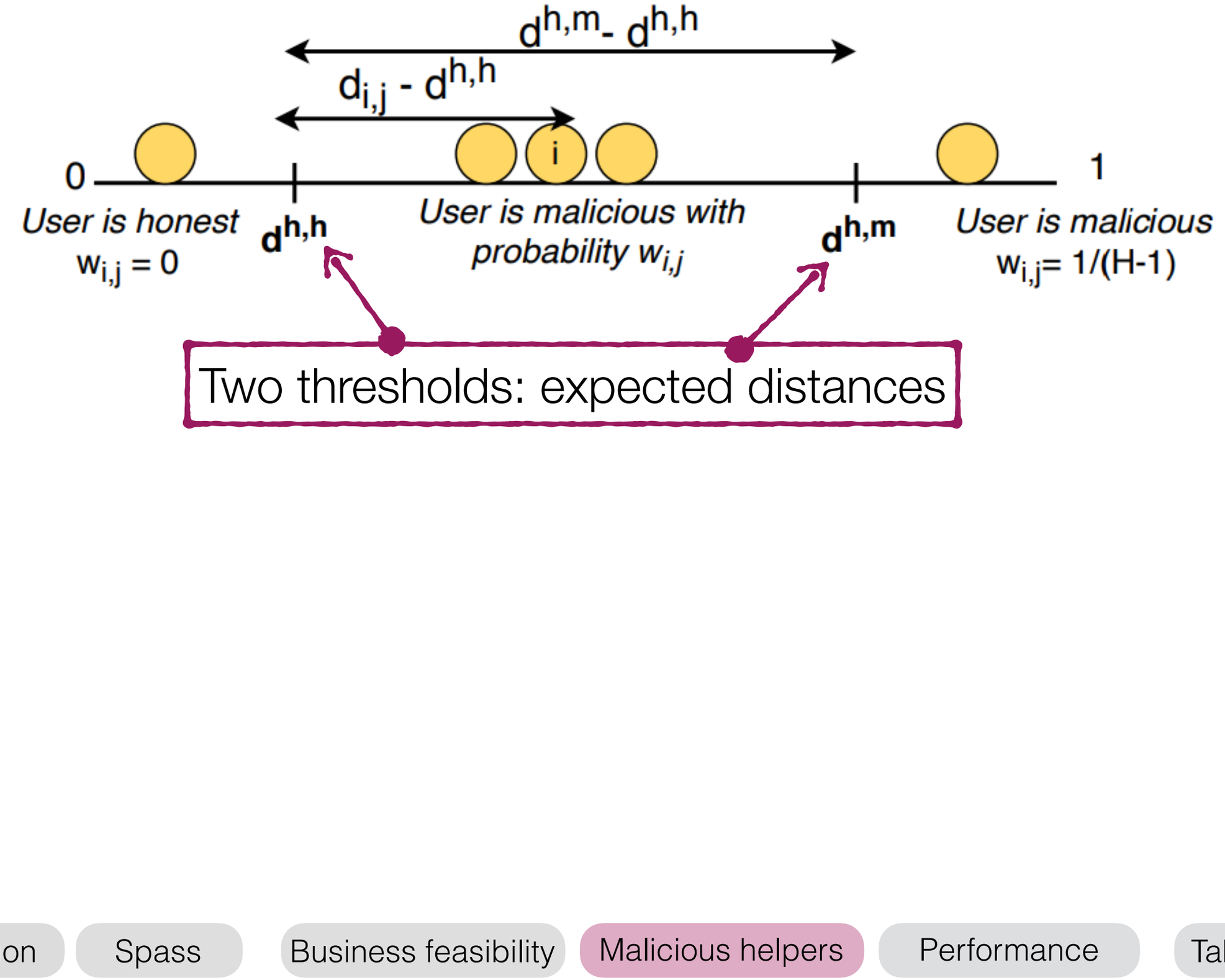
Two-threshold based voting (TTBV) for each helper



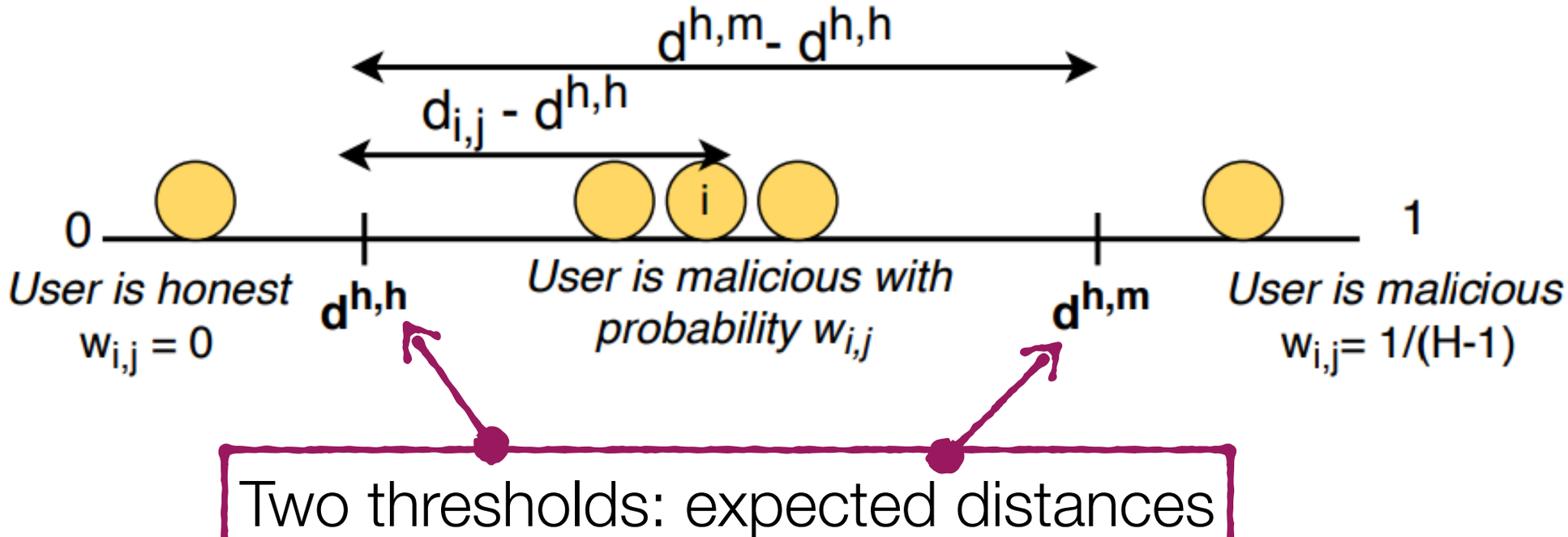
Two-threshold based voting (TTBV) for each helper



Two-threshold based voting (TTBV) for each helper

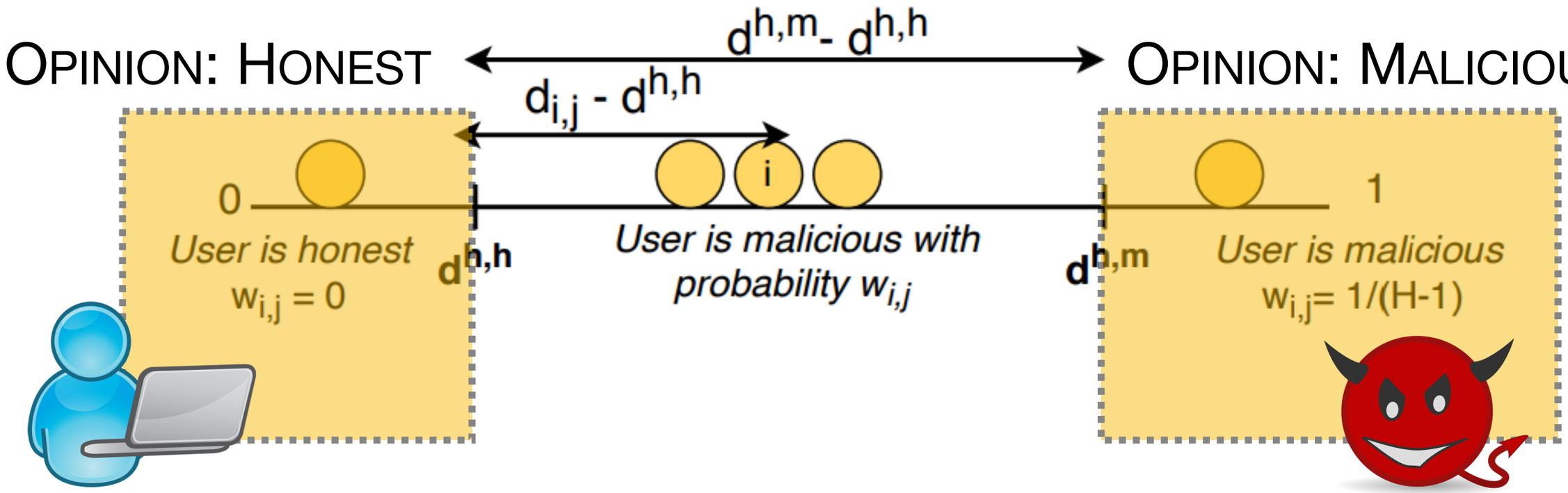


Two-threshold based voting (TTBV) for each helper



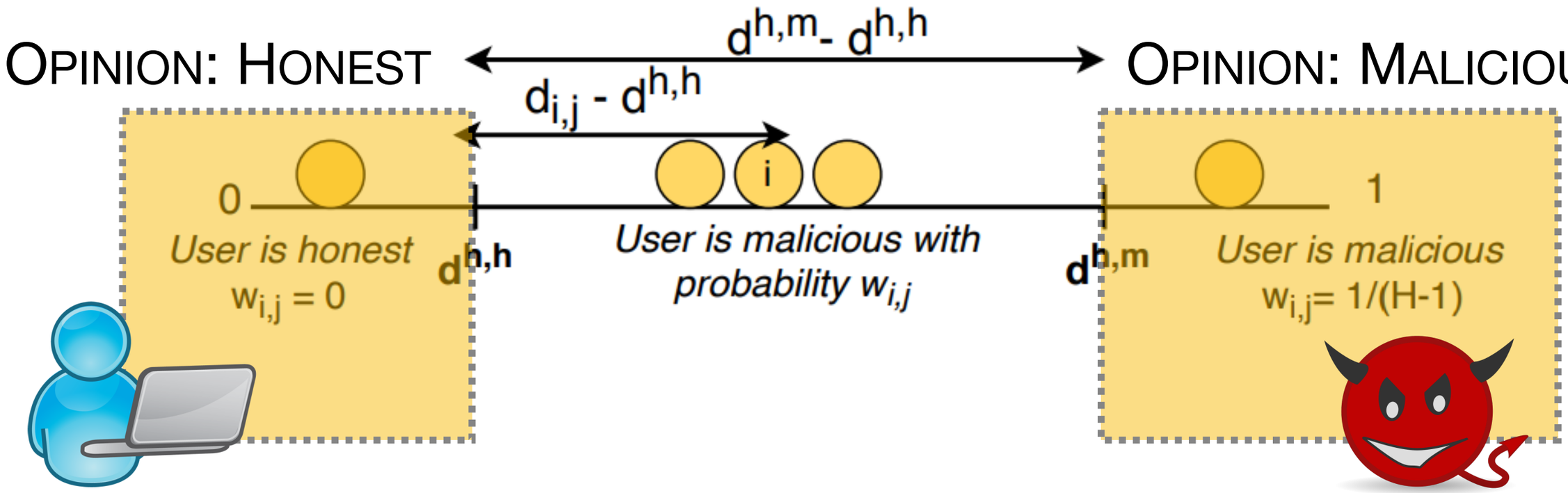
$$w_{i,j} = \begin{cases} 0 & \text{if } d_{i,j} \leqslant d^{h,h} \\ 1/(H-1) & \text{if } d_{i,j} \geqslant d^{h,m} \\ \frac{d_{i,j} - d^{h,h}}{(d^{h,m} - d^{h,h}) \times (H-1)} & \textbf{ow.} \end{cases}$$

Two-threshold based voting (TTBV) for each helper



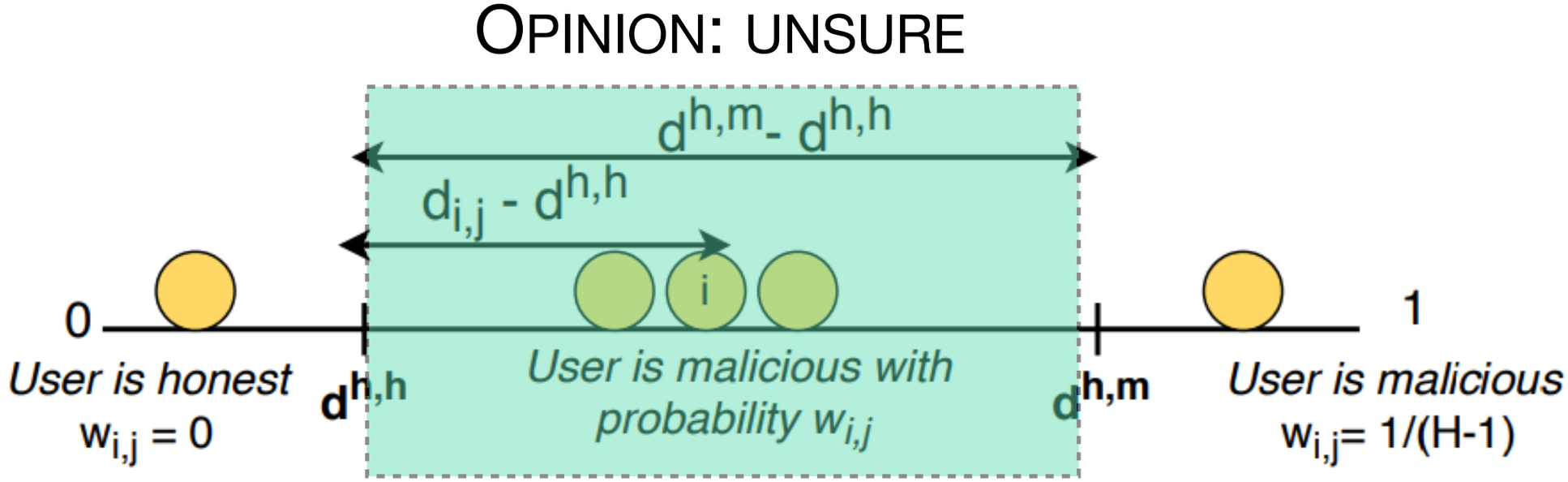
$$w_{i,j} = \begin{cases} 0 & \text{if } d_{i,j} \leqslant d^{h,h} \\ 1/(H-1) & \text{if } d_{i,j} \geqslant d^{h,m} \\ \frac{d_{i,j} - d^{h,h}}{(d^{h,m} - d^{h,h}) \times (H-1)} & \text{ow.} \end{cases}$$

Two-threshold based voting (TTBV) for each helper



$$w_{i,j} = \begin{cases} 0 & \text{if } d_{i,j} \leqslant d^{h,h} \\ 1/(H-1) & \text{if } d_{i,j} \geqslant d^{h,m} \\ \frac{d_{i,j} - d^{h,h}}{(d^{h,m} - d^{h,h}) \times (H-1)} & \textbf{ow.} \end{cases}$$

Two-threshold based voting (TTBV) for each helper

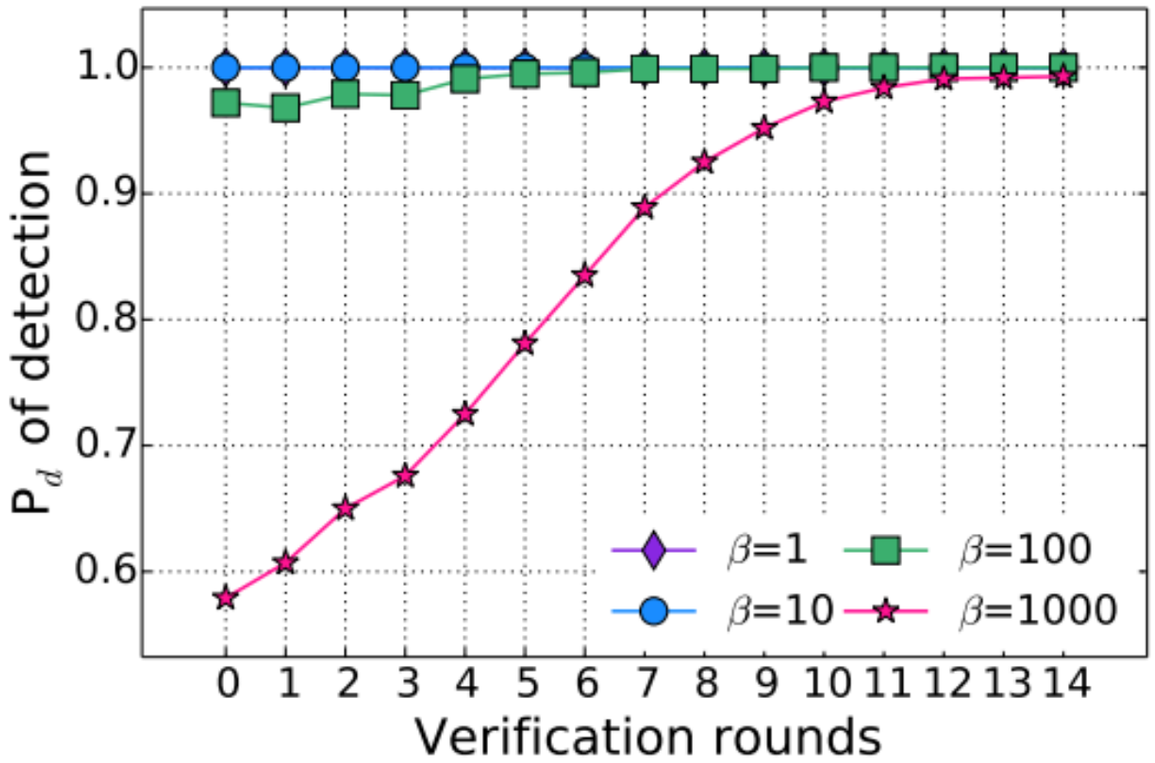


$$w_{i,j} = \begin{cases} 0 & \text{if } d_{i,j} \leqslant d^{h,h} \\ 1/(H-1) & \text{if } d_{i,j} \geqslant d^{h,m} \\ \dfrac{d_{i,j} - d^{h,h}}{(d^{h,m} - d^{h,h}) \times (H-1)} & \text{OW.} \end{cases}$$

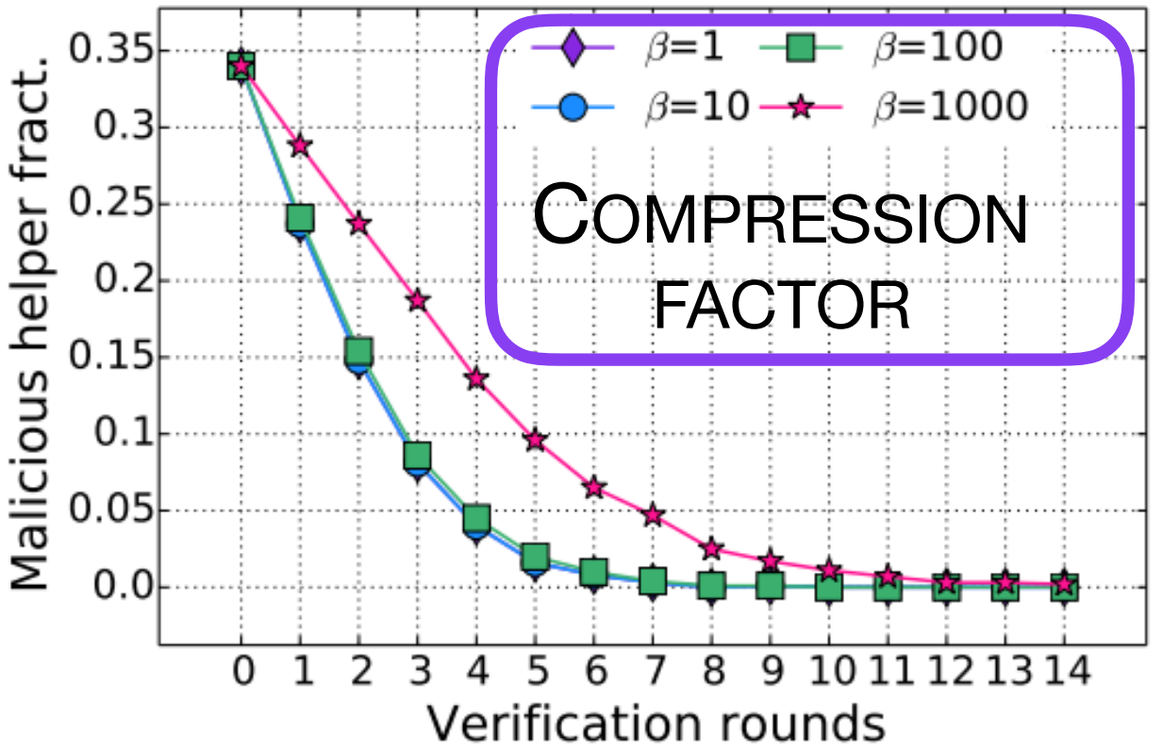
Performance analysis

- Python simulator
- Total 20 candidate helpers, 5 malicious, 8 helpers selected for sensing in every round
- Each round: 5000 time slots
- Honest Pd = 0.90, Ph=0.05, PU idle=0.6, Rs=1 Hz
- Impact of
    - (fraction of) malicious helpers
    - compression factor
- Accuracy of malicious helper detection
        - blacklisted honest helpers
        - time to detect all malicious helpers
- Accuracy of sensing
        - Pd and Pf
- Ethereum (Solidity) prototype
    - analysis of cost of functions

Accuracy of malicious helper identification



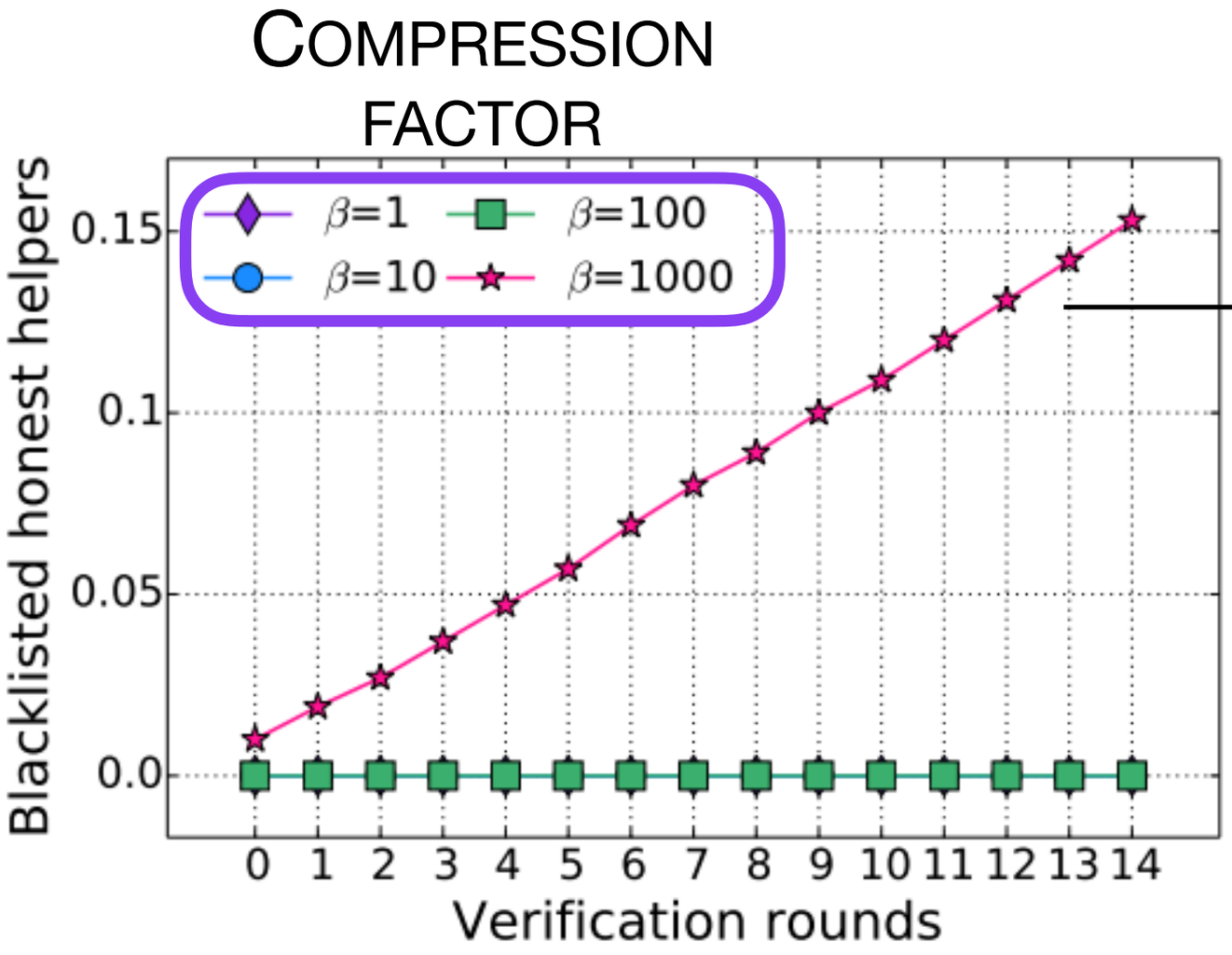
ACCURACY OF TTBV AS PD



MALICIOUS HELPERS IN A CONRACT

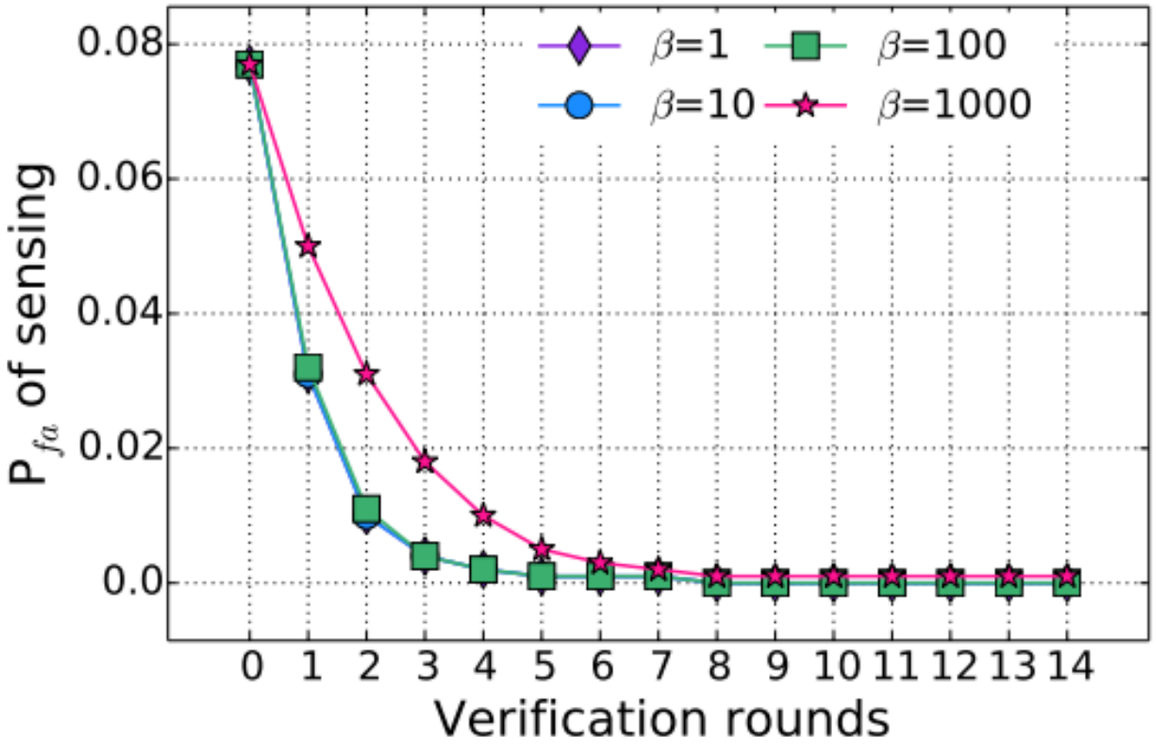
- Our algorithm can identify all malicious helpers after 6-7 rounds
- with increasing β (compression factor), the time to identifying all malicious helpers is longer

Blacklisting honest helpers must be avoided

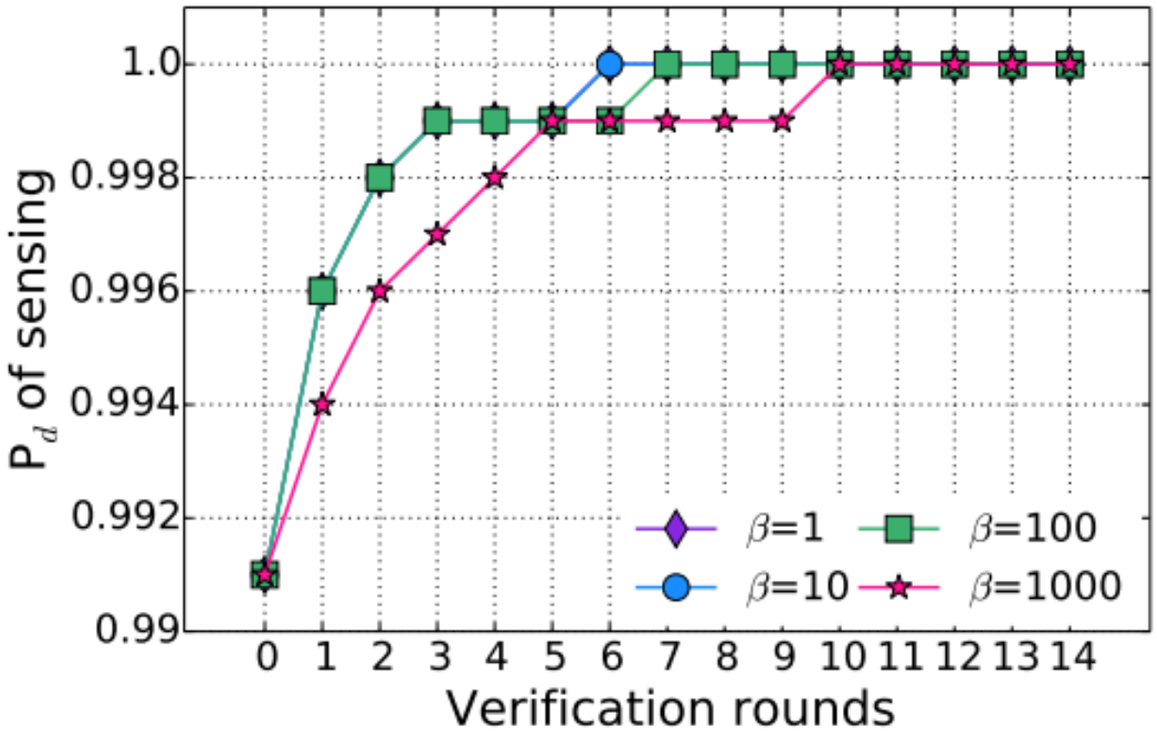


undesirable as it makes helpers hesitate to participate in Spass!

Sensing accuracy



(e) P_{fa} in sensing.



(f) P_d of spectrum sensing.

- Majority logic is robust to malicious helpers
- Hence, the sensing accuracy is not drastically affected

Spass prototype: Ethereum smart contract code

TABLE I
COST OF FUNCTION INVOCATION IN CONTRACT (H=HELPERS).

Function	Caller	Cyclic	TX (gas)	Ex. (gas)	€
<create contract>	SU	no	1638213	1223229	8.33
increaseFunds	SU	no	21579	307	0.06
init	SU	no	68630	46270	0.34
registerSensingHelper	H	no	178506	154994	0.98
waitForOtherHelpers	H	no	22995	1723	0.07
reportSensingData	H	yes	56814	32406	0.26
clearing	SU	yes	54372	32908	0.26
withdraw	H	no	19426	13154	0.10

Assumed 1 ETH=500$. The current price is 220$.

sensing rate R_S of 10Hz,
compression factor β of 100
verification round duration V of 15 min
15 Bytes of sensing data in each round

Source code is available under: https://github.com/zubow/Spass_contract

Take aways and future work

- Spass: smart contracts for spectrum sensing
- Trading in a distributed, trust-less environment
- Smart contracts are not free!
- Business feasibility analysis of Spass considering sensing and Ethereum smart contract costs
- Future/Ongoing work:
  - Realistic malicious helper models
  - Pricing/payment strategies of helpers and helper selection
  - Lossless compression

Take aways and future work

- Spass: smart contracts for spectrum sensing
- Trading in a distributed, trust-less environment
- Smart contracts are not free!
- Business feasibility analysis of Spass considering sensing and Ethereum smart contract costs
- Future/Ongoing work:
  - Realistic malicious helper models
  - Pricing/payment strategies of helpers and helper selection
  - Lossless compression

Thank you!

Suzan Bayhan, Anatolij Zubow, and Adam Wolisz
https://suzanbayhan.github.io/

Technische Universität Berlin