

Decentralized Finance II

Ronghui Gu

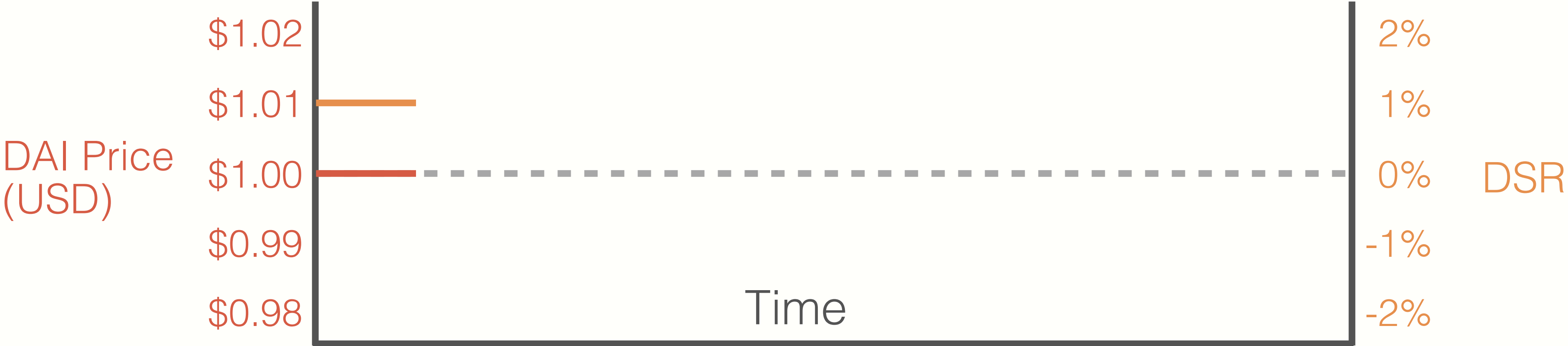
Fall 2024

Columbia University

Course website: <https://verigu.github.io/6998Fall2024/>

0 **NO Financial Advice!!!**

Synthetic Stablecoins — Stabilization



The stability fee and DSR are raised when DAI is trading below \$1 (to discourage borrowing and encourage DAI holding), and lowered when DAI is trading above \$1

Background

A blockchain cannot access **data outside of its state** (e.g. ETHUSD price, the weather today, content at a URL, etc.)

Complex use cases require non-native data

- Finance: prices, insurance
- Random number generation
- Blockchain interoperability: bitcoin headers on Ethereum
- IoT: temperature, geolocation data etc.

How do you import non-native data to a blockchain? **Oracles!**

Price Oracles in DeFi

Stablecoins and synthetics

- Liquidations
- Interest rates
- Redemptions

Lending

Derivatives

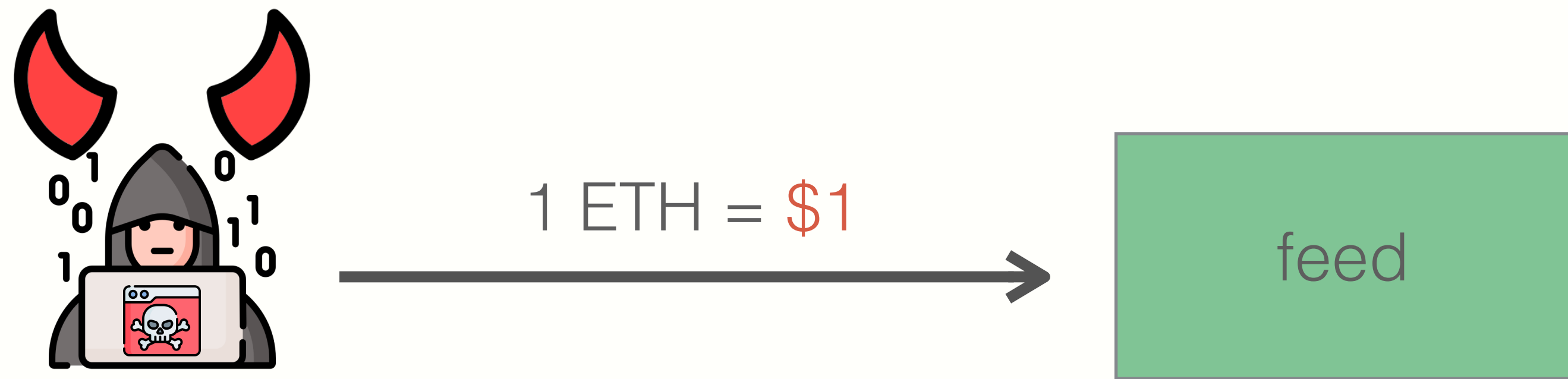
Trusted Signer

```
contract Oracle {
    address oracle;
    uint256 public ethusd;

    constructor() {
        oracle = msg.sender;
    }

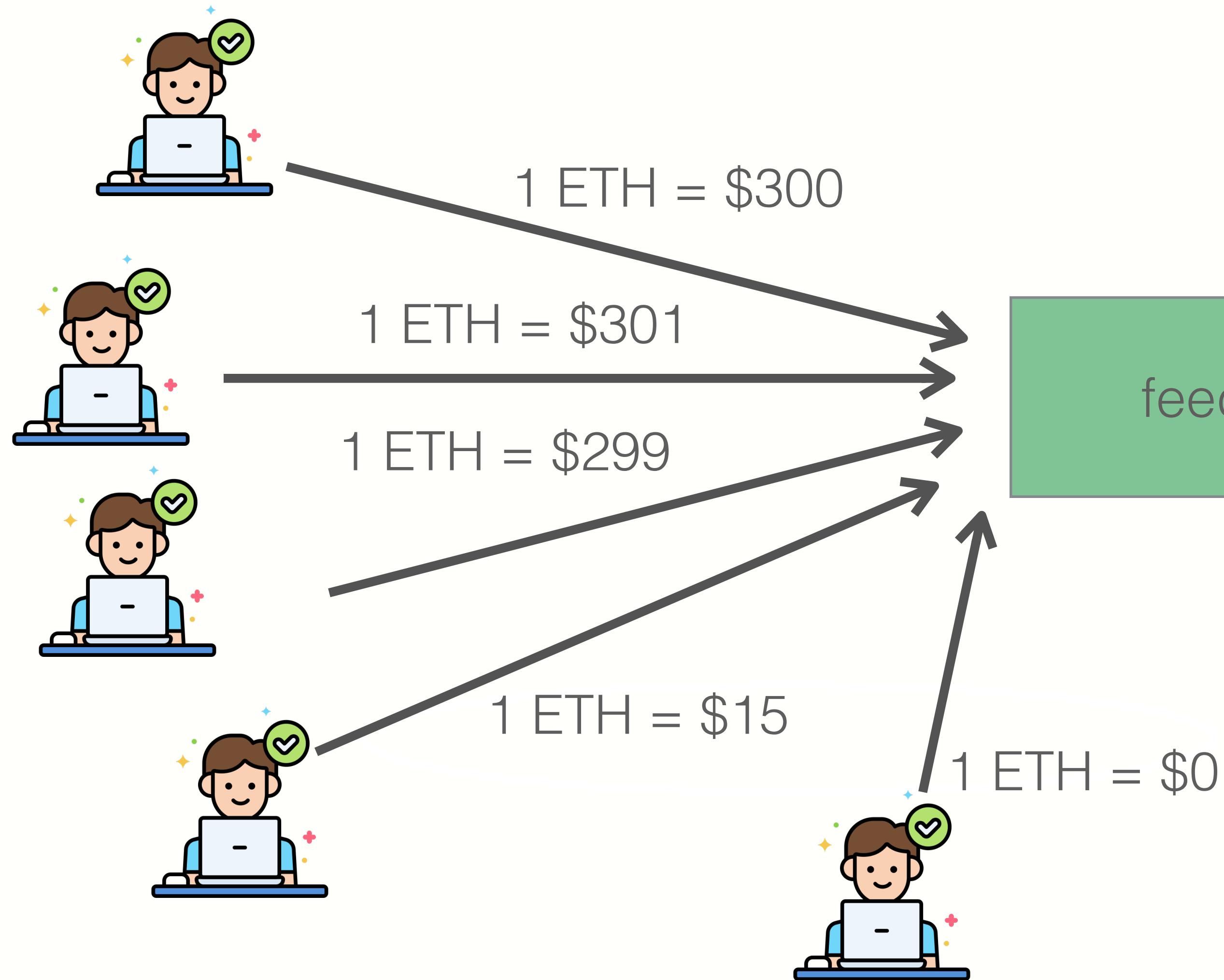
    function update(uint256 ethusd) external {
        require(msg.sender == oracle, "auth error");
        ethusd = _ethusd;
    }
}
```

Trusted Signer



Corrupt signer can manipulate!

M of N Oracles

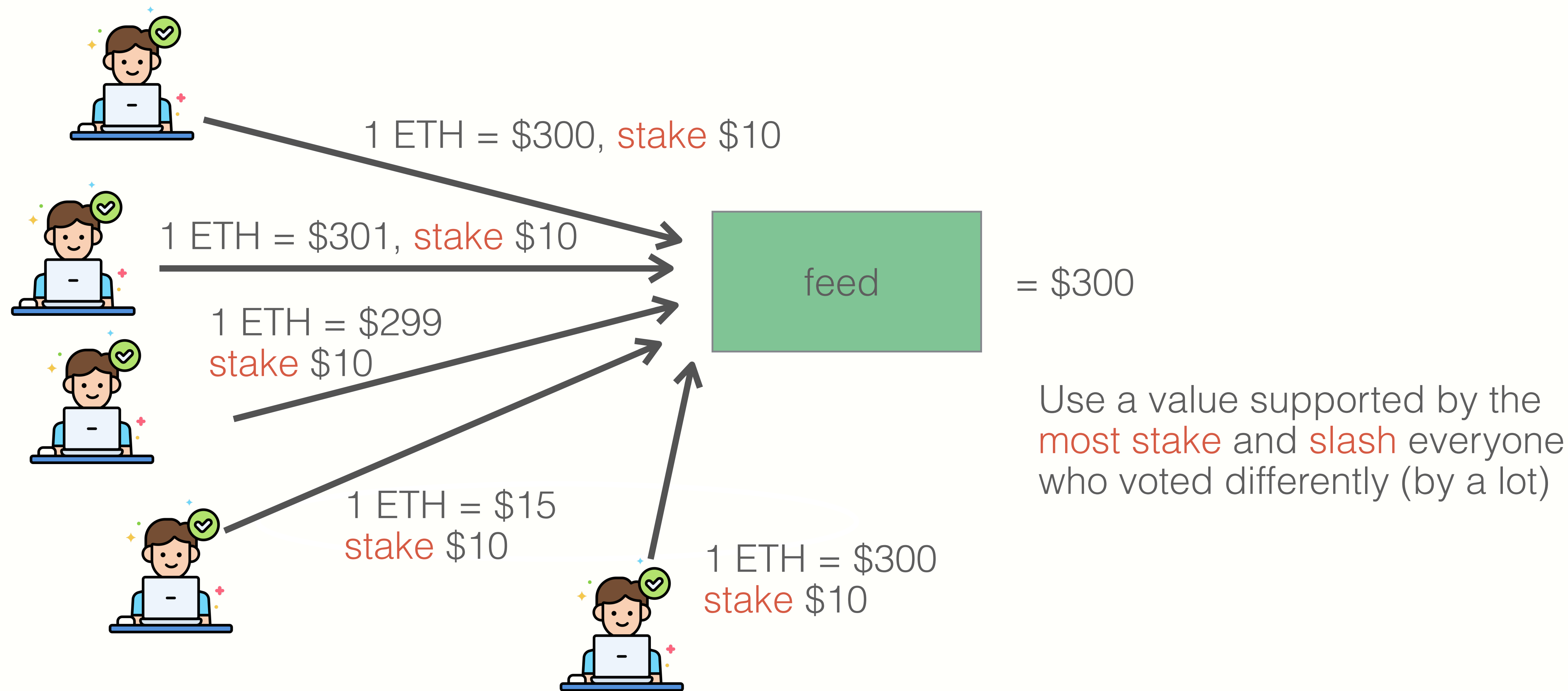


In general, uses a **consensus** protocol to resolve discrepancies

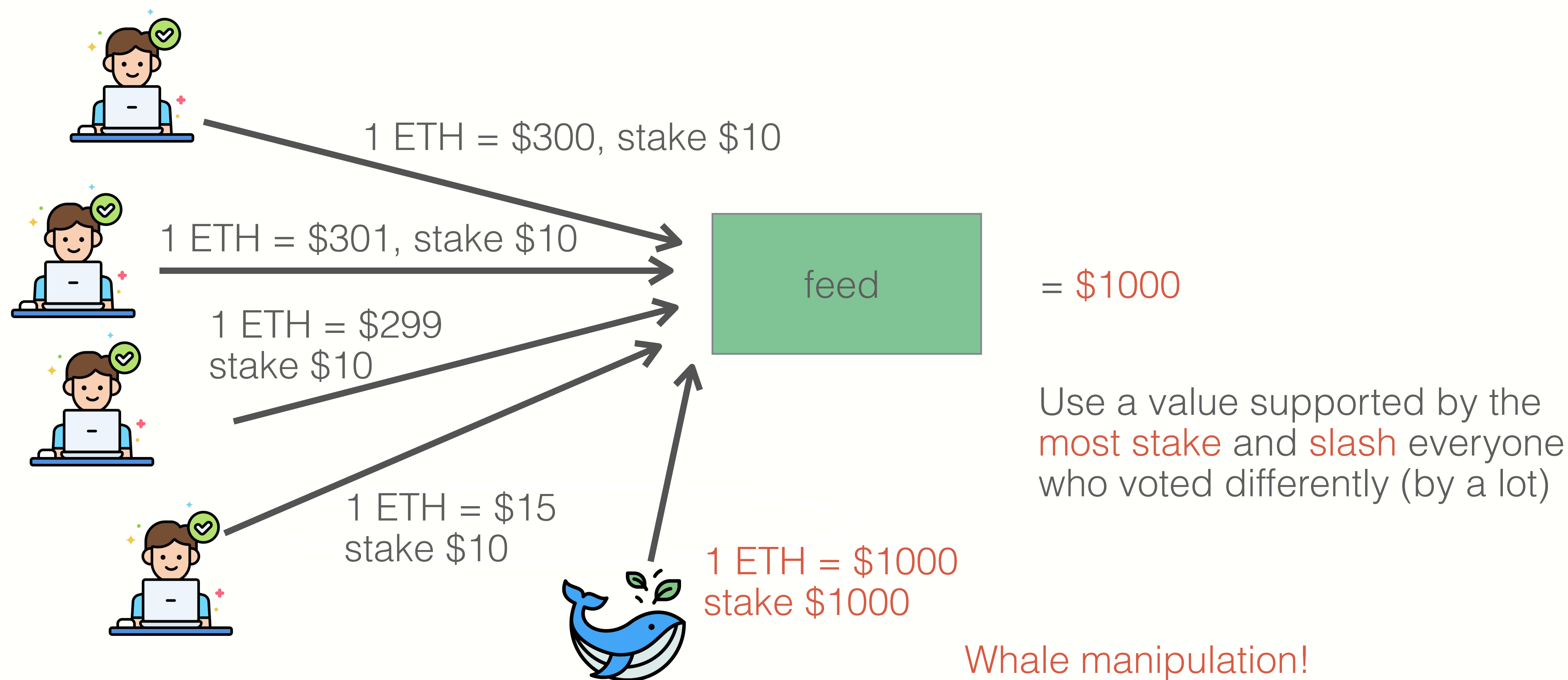
If **3 of 5** have published a value, take the **median** (here: \$300)

Vulnerable against collusion

M of N Oracles



M of N Oracles

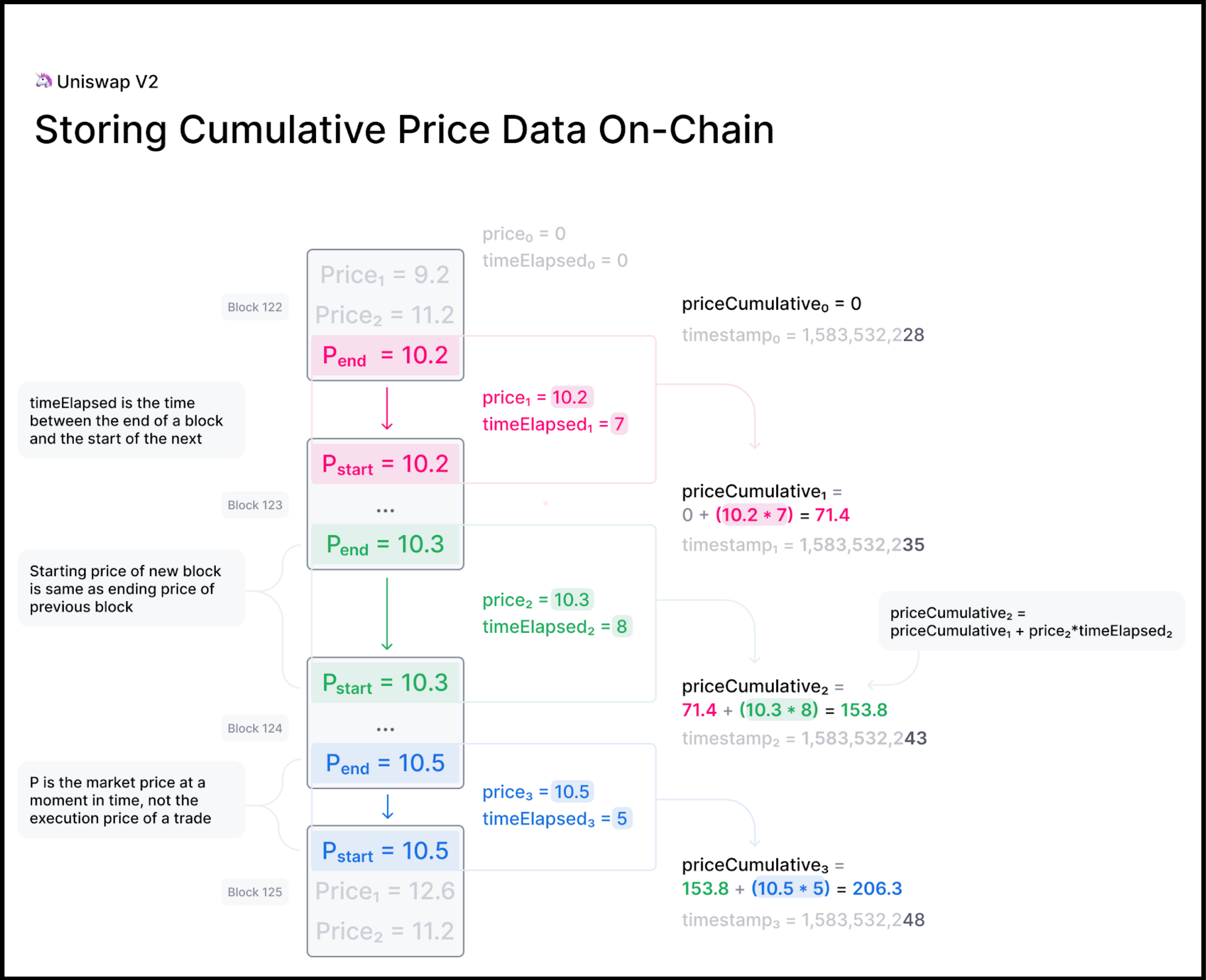


On-chain Oracles

For on-chain assets, we can use DEXes (auctions, orderbooks, or AMMs) as **price oracles**!

Sandwich attacks!

Uniswap — time weighted average

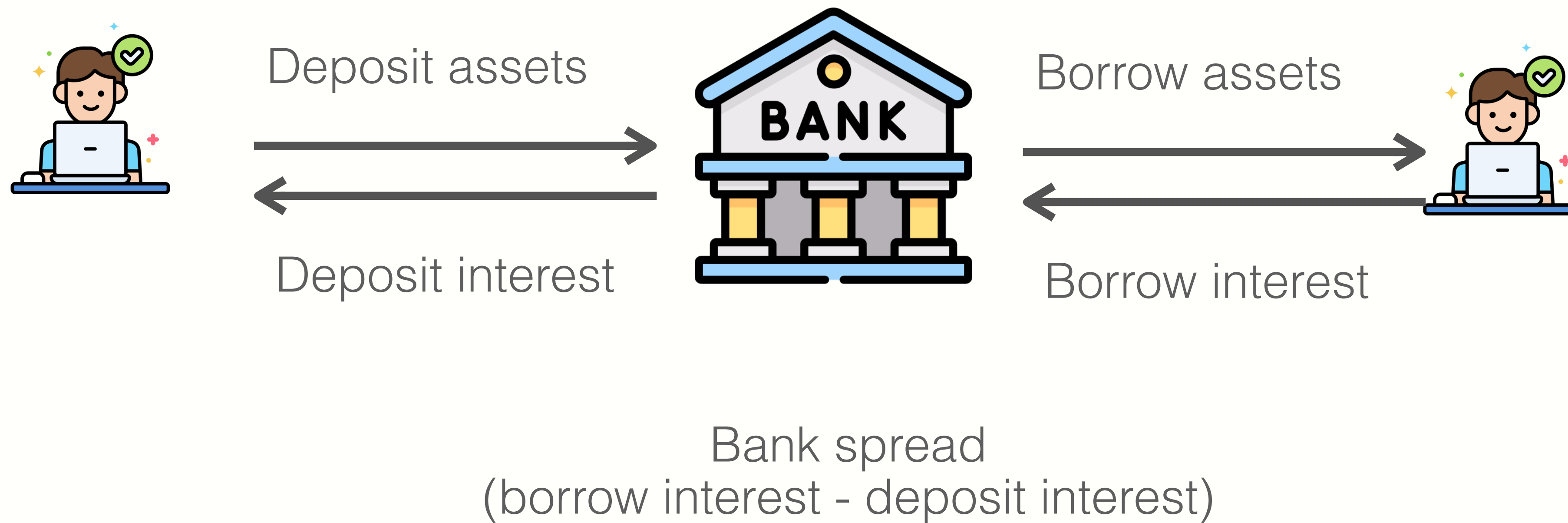


Recap: mapping the oracle design space

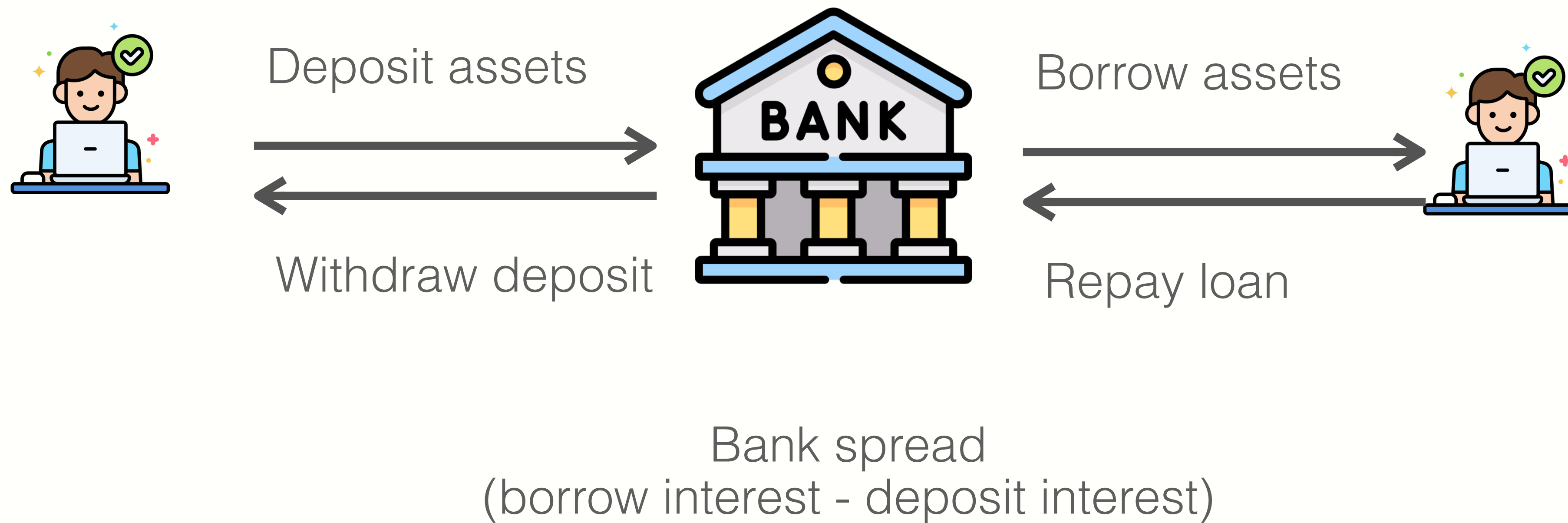
	Decentralized	Freq/Accuracy	Corruption cost
1 signer	Low	High	Low
M of N signer	Medium	High	Medium
On-chain oracle	High	High	Low
Uniswap TWAP	High	Configurable	High

2 DeFi Lending Systems

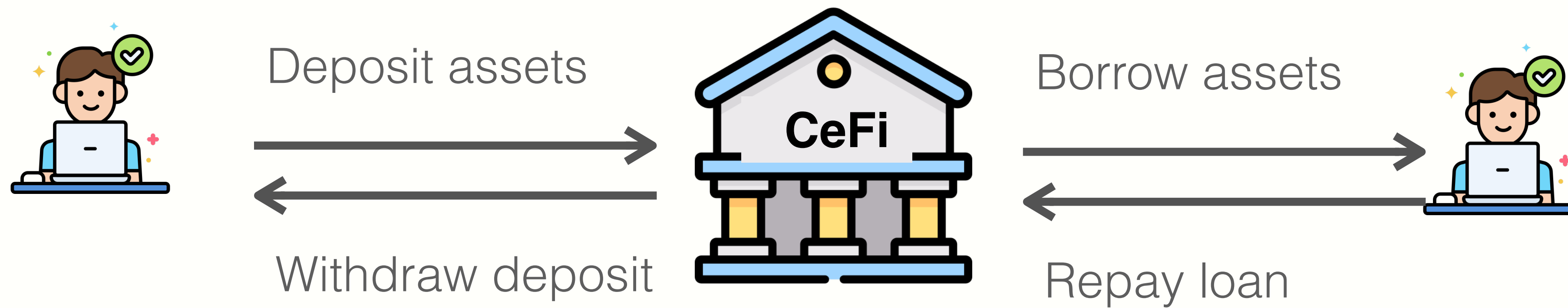
The role of banks



The role of banks



The Role of Crypto CeFi Lending

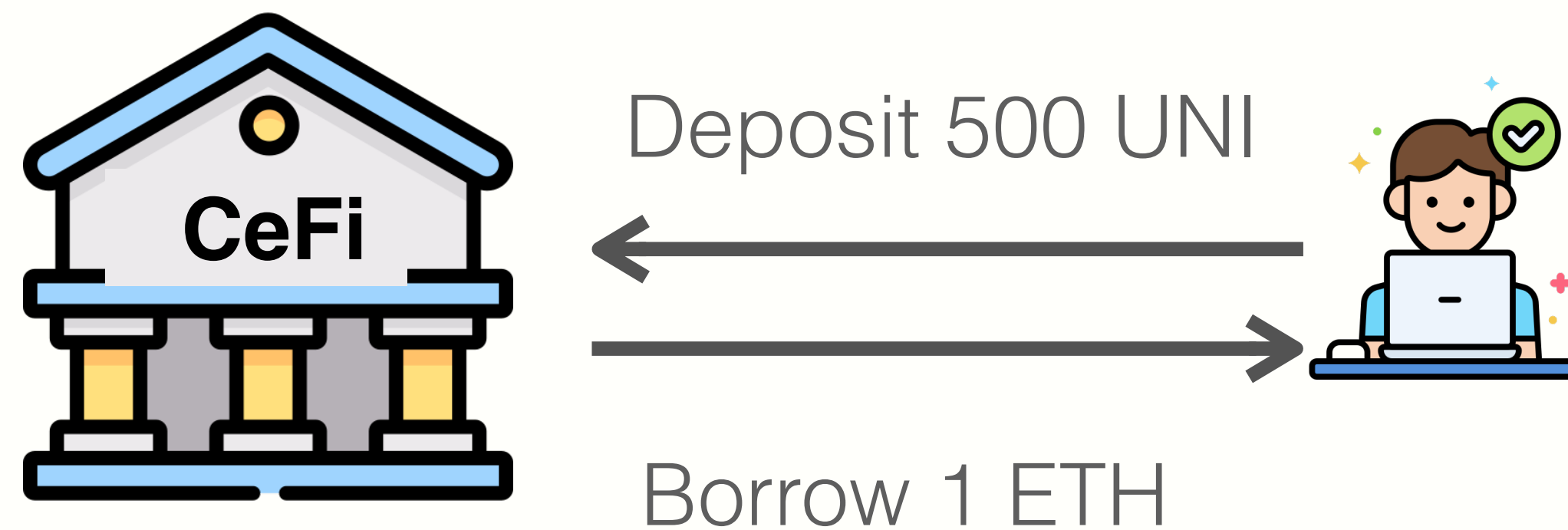


The Role of Crypto CeFi Lending

CeFi's concern: what if Bob **defaults** on loan?

Solution: require user to **lock up** collateral

lender needs to **liquidate** before $\text{value}(\text{debt}) > \text{value}(\text{collateral})$



Why Borrow ETH?

If Bob has collateral, why can't he just **buy** ETH?

- Bob may need ETH (e.g., to buy in-game Axies), but he might not want to **sell** his collateral (e.g., an NFT)
- As an investment strategy:
using UNI to borrow ETH gives Bob exposure to both

The problem with CeFi lending

Users must **trust the CeFi institution:**

- Not to get hacked, steal assets, or miscalculate
- This is why traditional finance is regulated

Interest payments go to the exchange, not liquidity provider

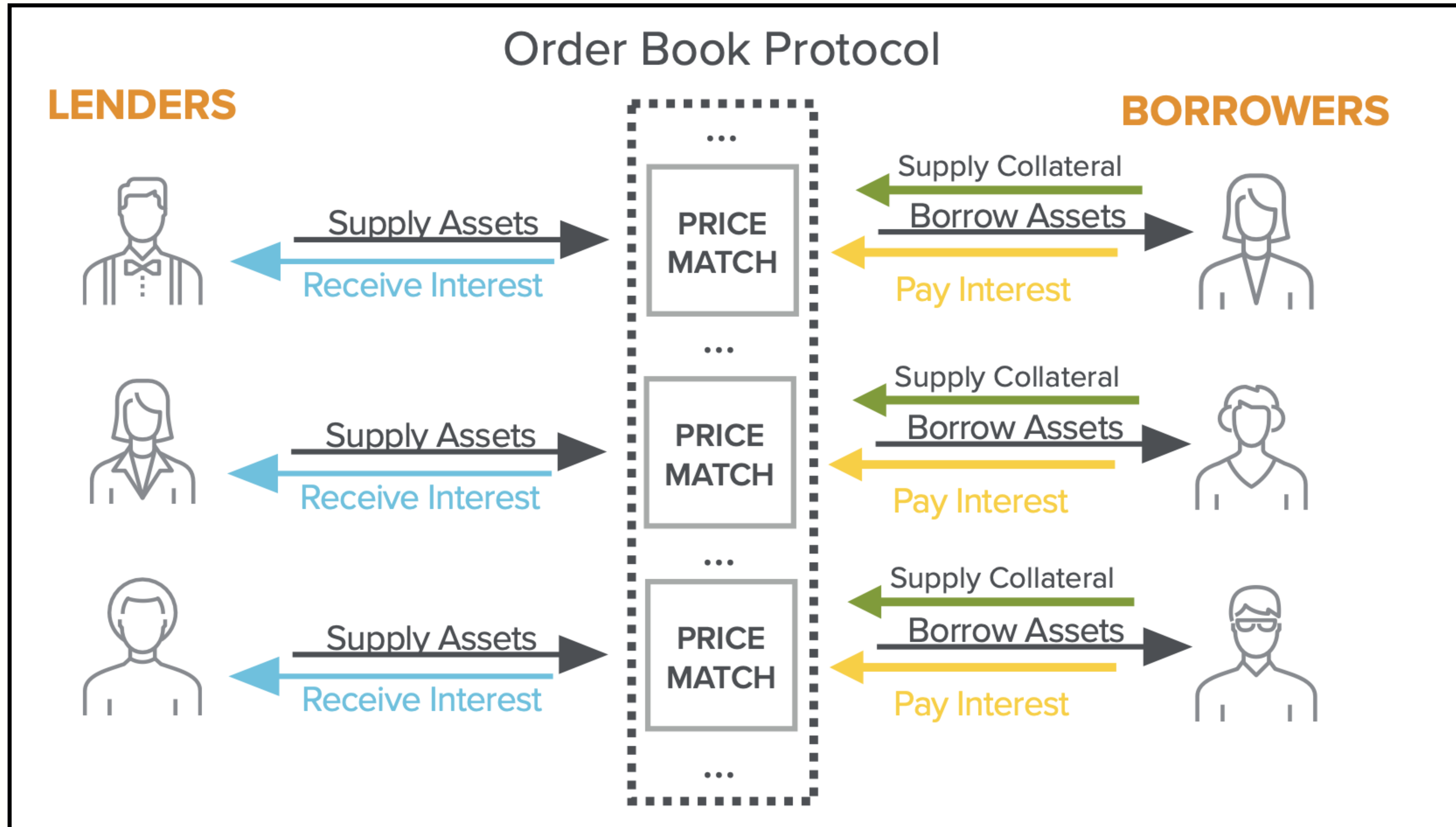
CeFi fully controls spread (borrow interest – deposit interest)

DeFi Lending

Can we build an **on-chain** lending Dapp?

- no central trusted parties
- code available on Ethereum for inspection

A first idea: an order book Dapp



Problems

Computationally expensive:

matching borrowers to lenders requires **many** transactions per person

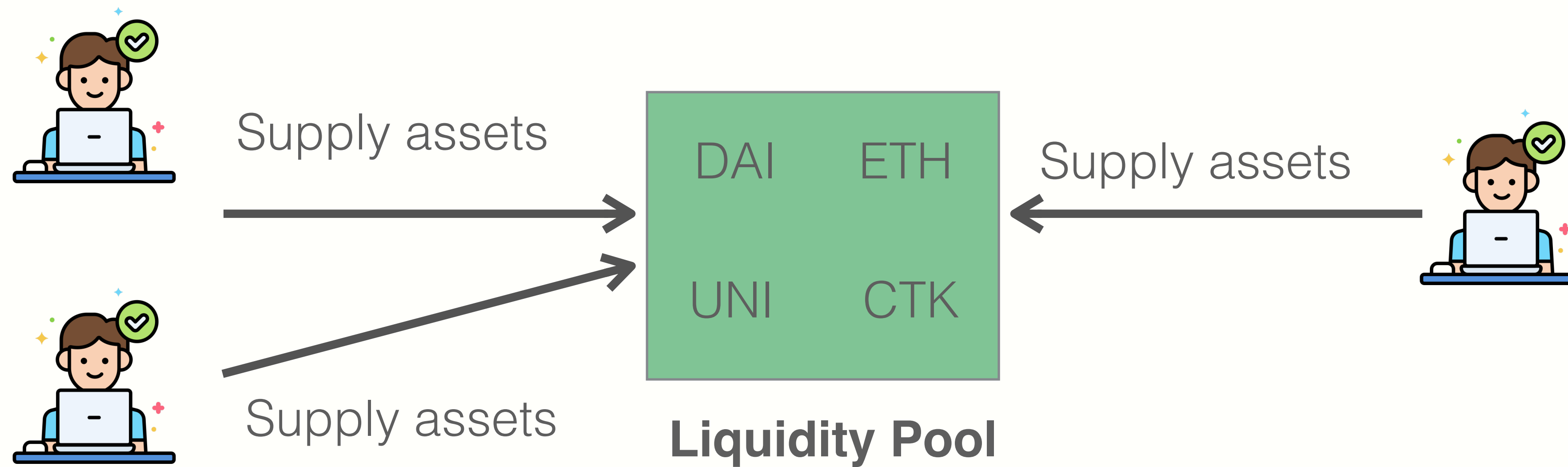
Concentrated risk:

lenders are exposed to their **direct** counterparty defaulting

Complex withdrawal:

a lender must **wait** for their counter-parties to repay their debts

A better approach: liquidity pools

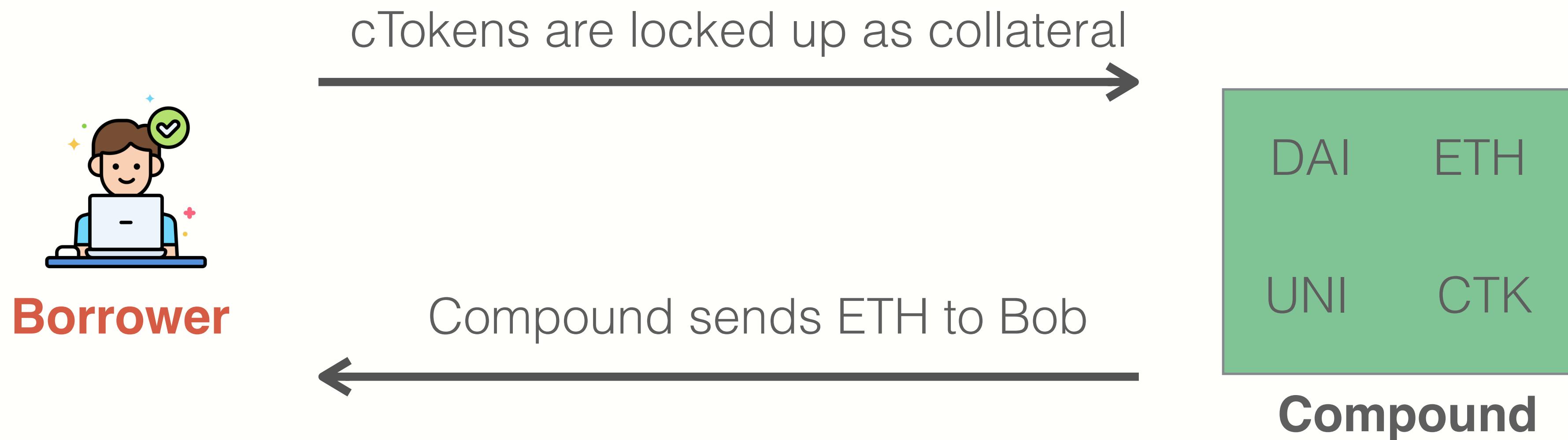


Example: Compound cTokens



Value of X, Y, Z is determined by an **exchange rate**

Example: Compound cTokens



Bob's accrued interest increases ETH/cETH exchange rate

=> **benefit** cETH token holders (ETH liquidity providers)

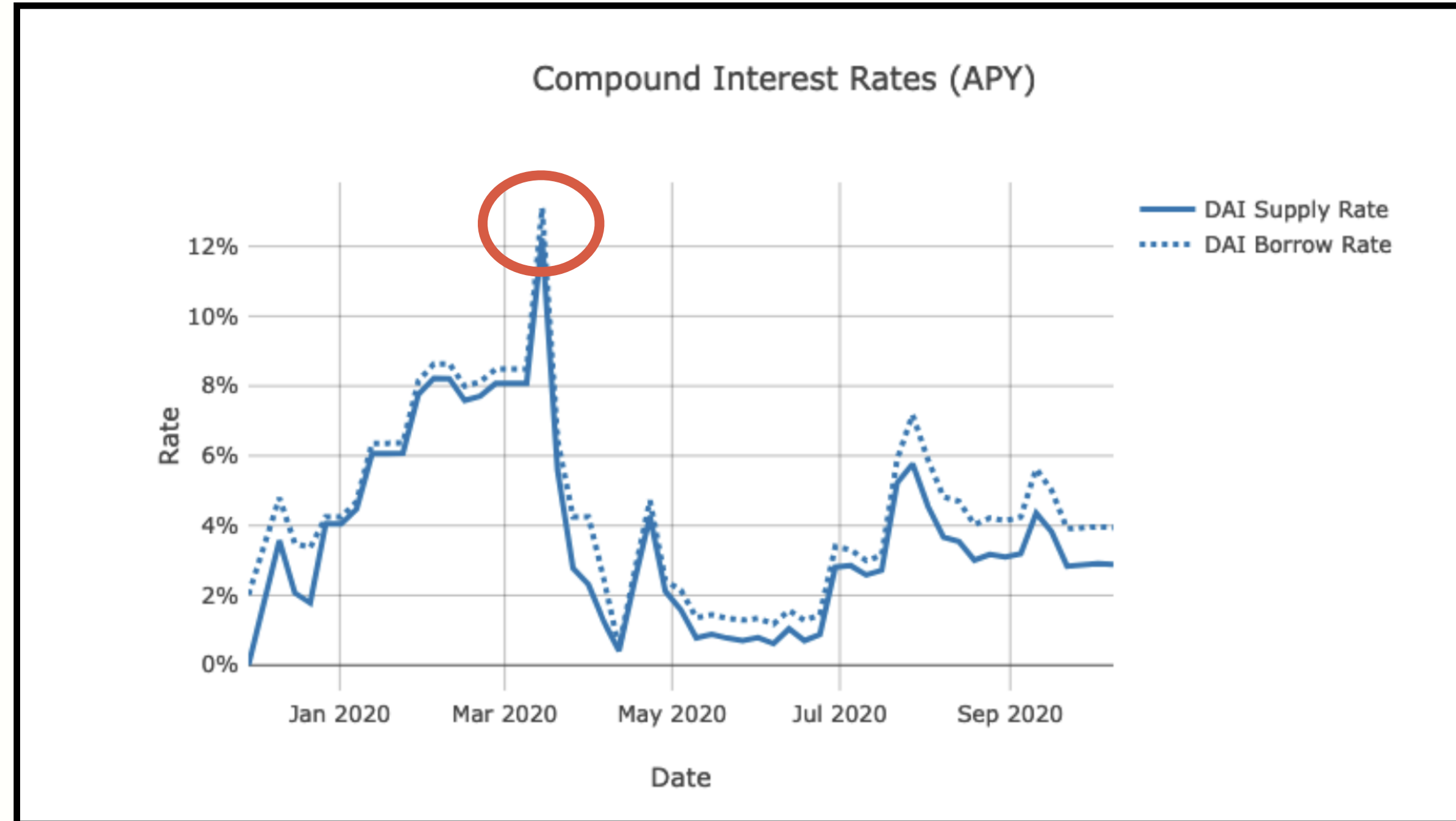
Liquidation risk

Demand for DAI spikes

=> price of DAI spikes

=> user's debt shoots up

=> liquidation



To use Compound, borrowers must **constantly monitor** APY and **repay loans** if APY goes too high

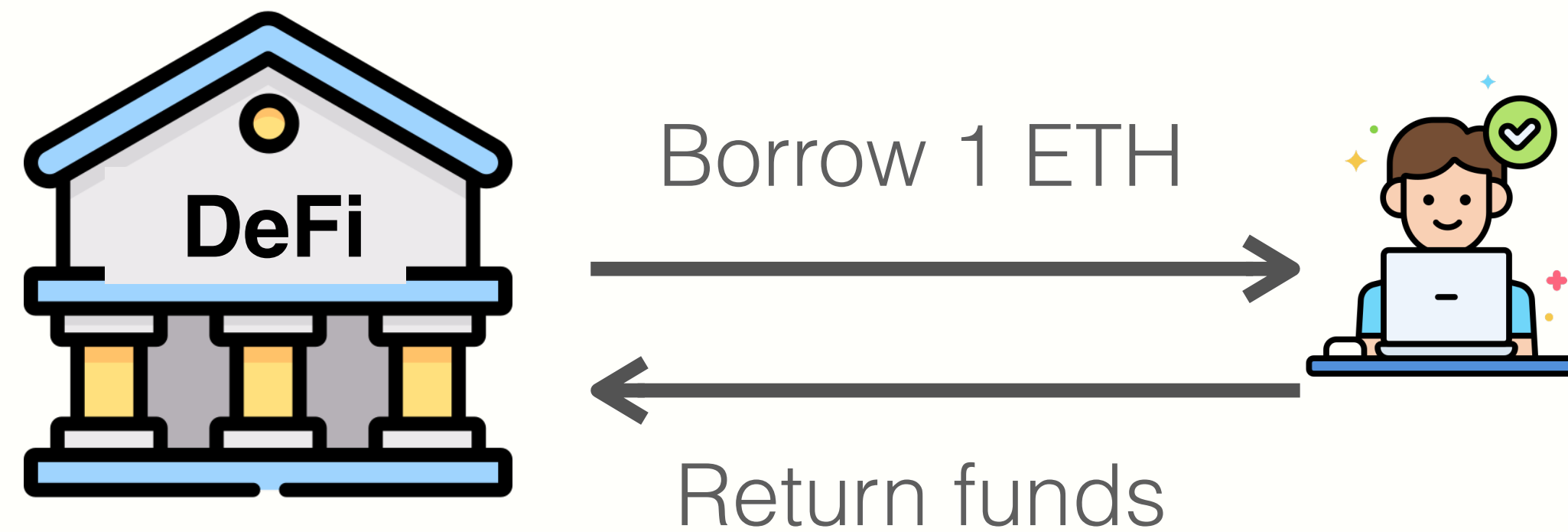
3 Flash Loan

What is a flash loan?

A flash loan is taken and repaid in a **single** transaction

=> **Zero** risk for lender

=> Borrower needs **no** collateral



Tx is valid only if funds are returned in the same Tx

Use cases

Risk free arbitrage

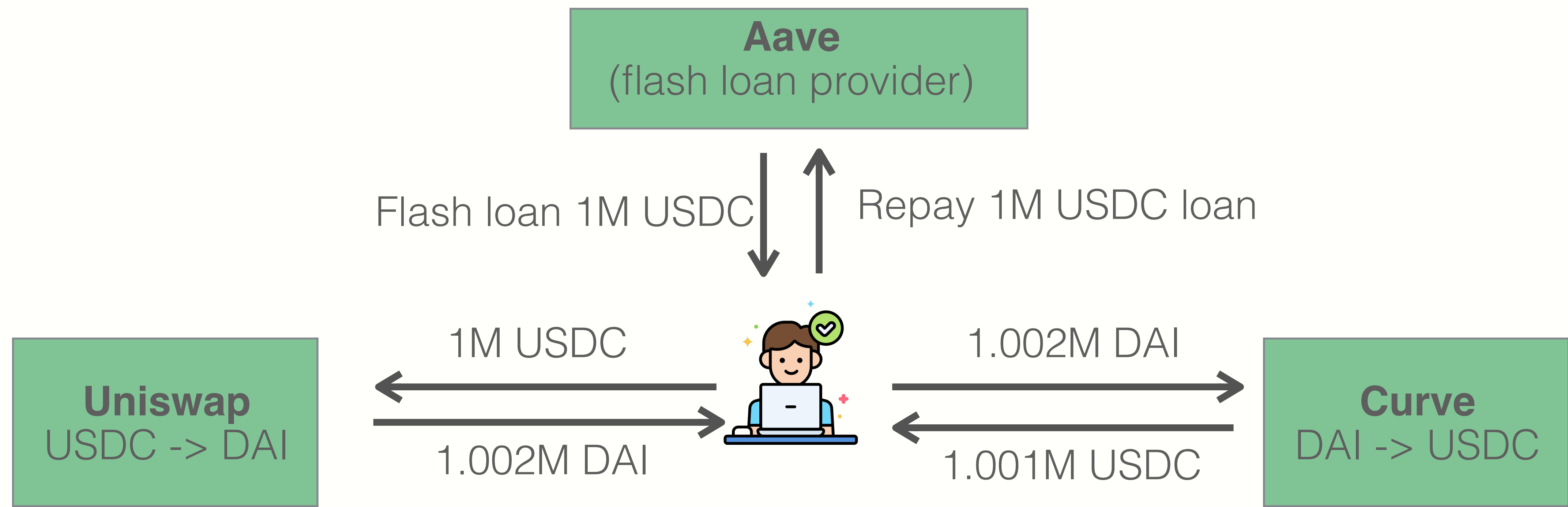
Collateral swap

DeFi attacks: price oracle manipulation

...

Risk free arbitrage

Bob finds a USDC/DAI price difference in two pools



Bob keeps **0.001M** USDC

All in a single transaction!

Aave v1 implementation

```
function flashLoan(address _receiver, uint256 _amount) {  
    ... // transfer funds to the receiver  
    core.transferToUser(_reserve, userPayable, _amount);  
  
    // execute action of the receiver  
    receiver.executeOperation(_reserve, _amount, amountFee, _params);  
  
    ... // abort if loan is not repaid  
    require(availableLiquidityAfter == availableLiquidityBefore.add(amountFee),  
            "balance inconsistent");  
}
```

Use cases

Risk free arbitrage

Collateral swap

DeFi attacks: price oracle **manipulation**

...

4 DAO

Decentralized orgs (DAO)

What is a DAO?

- A Dapp deployed on-chain at a specific address
- Anyone (globally) can send funds to DAO treasury
- Anyone can submit a proposal to DAO
⇒ participants **vote**

Decentralized orgs (DAO)

Creating a DAO is quite simple
cheaper than creating a real-world U.S. partnership

Example DAOs:

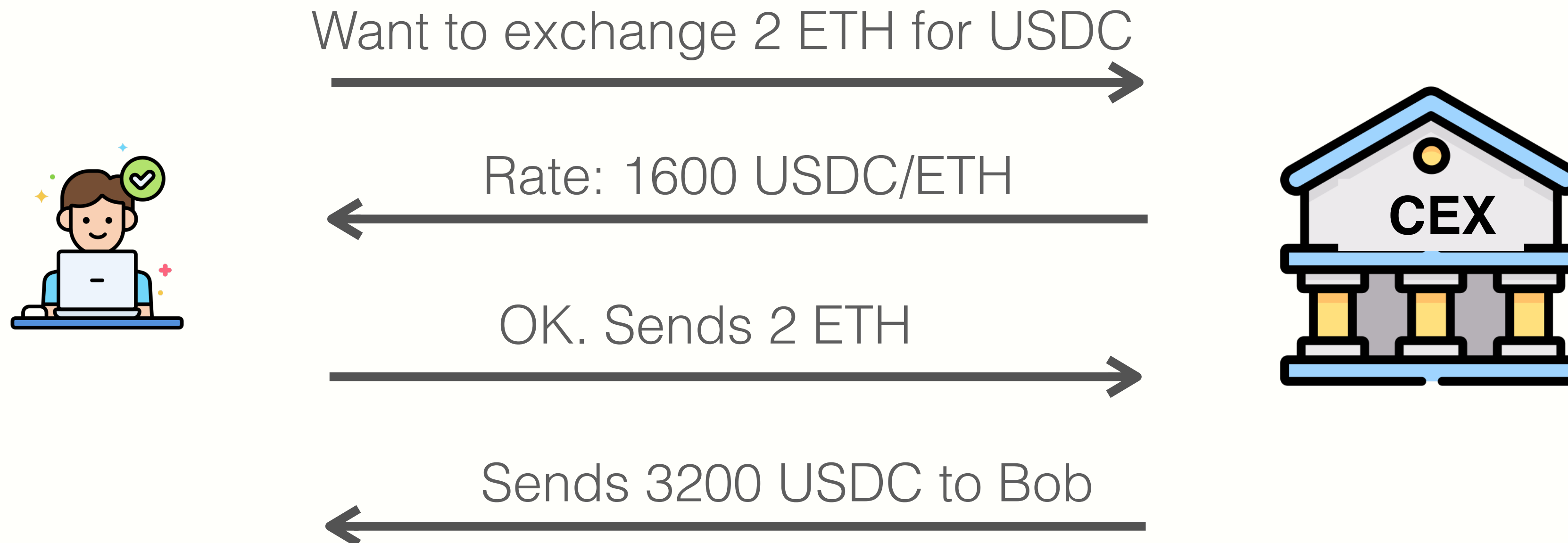
- **PleasrDAO**: invests in NFTs
- **Gitcoin**: DAO to fund open source projects
- **Investment DAOs**: many

5 DEX

What is an exchange?

An exchange: users to **convert** one token to another (e.g., USDC -> ETH)

- What is the exchange **rate**?
- How to **connect** sellers and buyers?



Problems

How is exchange rate **determined**?

- By **supply** and **demand** at the exchange (not transparent)
- Competition with other exchanges (bad user experience)

Security: What if exch. takes Bob's 2 ETH, but never sends USDC?

Censorship: What if exchange refuses to do business with Bob?

A more trusted solution: DEX

What is a DEX?

- a marketplace where transactions occur directly between participants, **without** a trusted intermediary

Properties:

- **Programmable:** can be used as a service by other contracts
- **Transparent:** code is available for everyone to see
- **Permission-less:** anyone can use
- **Non-Custodial**

How to build a DeX?

First idea: on-chain order book

- Liquidity providers place buy/sell orders on chain
- Users fill them on chain

Problem: gas inefficient

- Orders cost gas: when placed, when filled, when canceled.
- Matching buy orders to sell orders takes lots of gas

How to build a DeX?

Next idea: off-chain order book

- Liquidity providers sign buy/sell orders off chain
 - Post orders on a centralized web site
- User signs an order it wants to fill and submits it on chain.
- **Examples:** 0x Protocol, OpenSea

Problem: order book is not accessible to contracts (dAPPs)

How to build a DeX?

Next next idea: Automated Market Maker (AMM)

- Liquidity providers deposit assets into an on-chain pool
- Users trade with the on-chain pool
 - exchange rate is determined algorithmically
- **Examples:** Uniswap, Balancer, Bancor

Benefits: Gas-efficient, accessible to contracts, easy to bootstrap

6

Discussion Session

How to build a product?

