

Lecture 6

—

DeFi & Decentralized Exchanges

DeFi Overview

TradFi & current ecosystem

TradFi - Traditional Finance



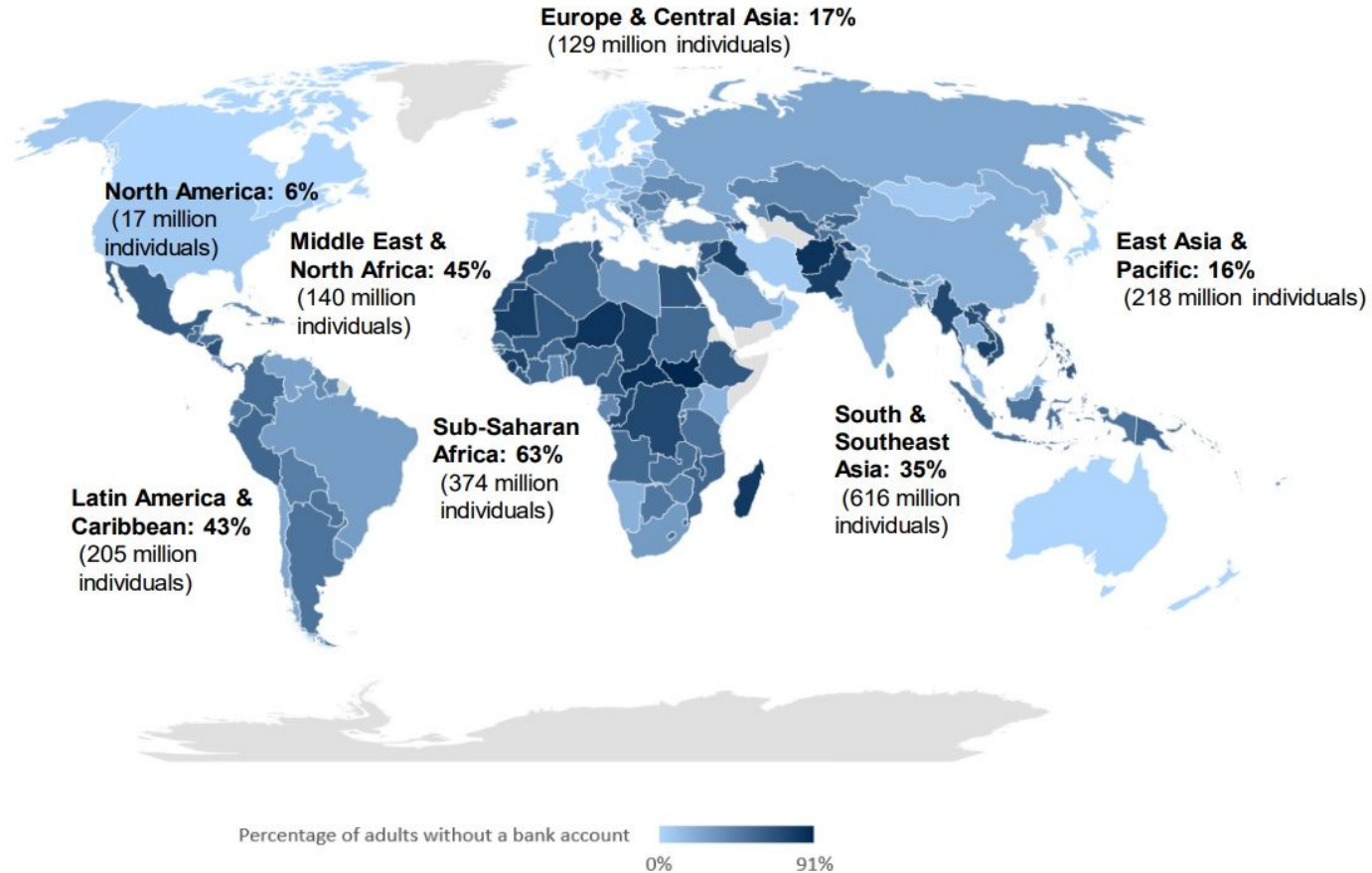
A **centralized** trusted authority that enables trustless transactions.

- Provides assurance of value
- Enforces compliance
- Fights fraud

Drawbacks of TradFi

- Subject to (sometimes very high) fees
 - Account fees and interest fees
 - Remittances for immigrants
- Money becomes political
 - Sanctions during the Ukraine crisis
 - Taxes, treaties for certain group
- Transfer speed
 - SEPA zone - used to be 48 hours, now 24 hours
 - International times???
- Monolithic
 - Many still running on Windows 95. programming in COBOL
 - Extremely difficult to change
- Exclusionary
 - r/wallstreetbets fights back - [link](#)
 - Millions of unbanked left behind by the system

Proportion of adult population without access to financial services

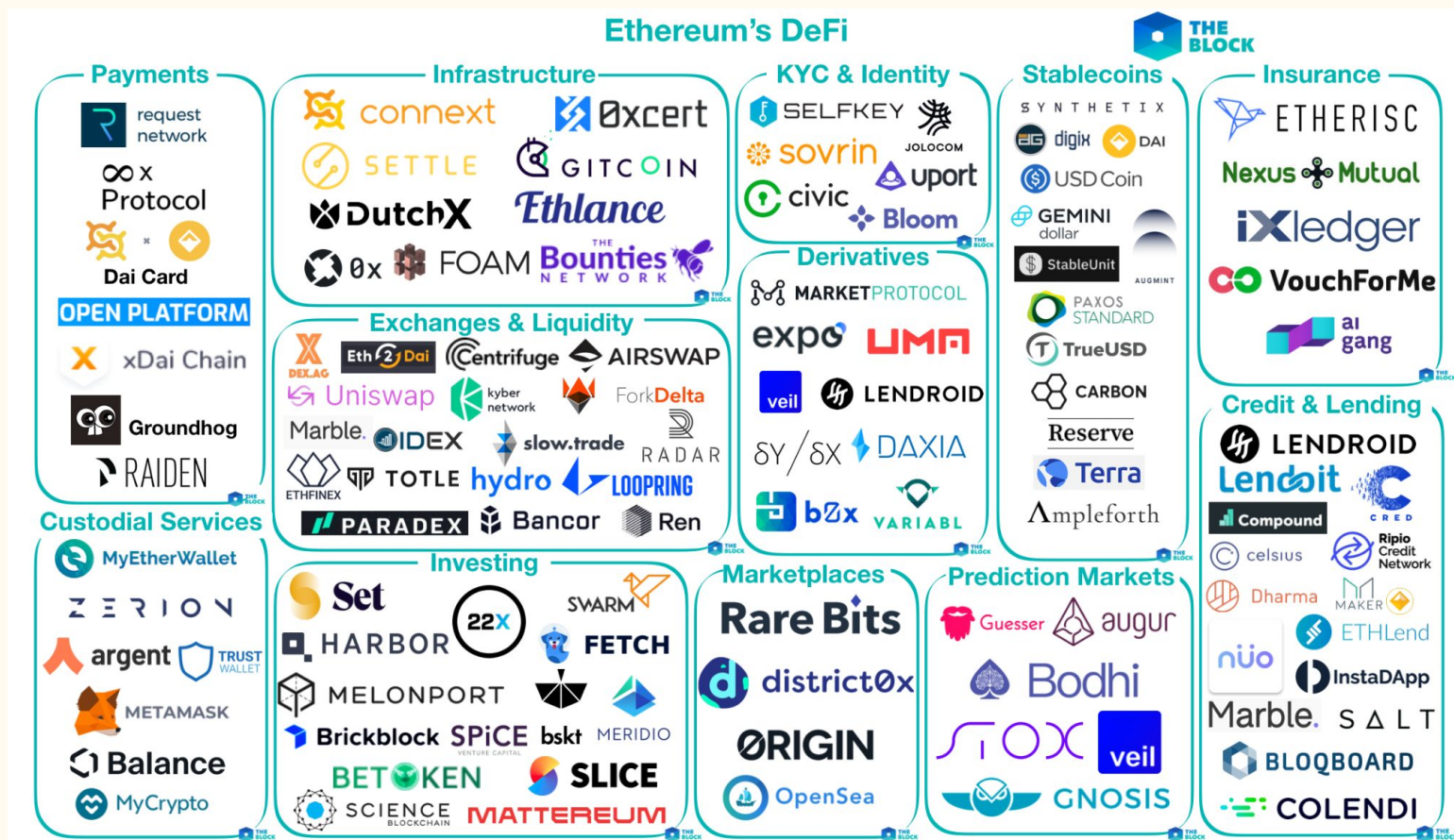


Decentralized Finance - What is DeFi?

A group of financial software products and services that:

- Has no central authority. Peer to peer to and completely governed by code
- Offers traditional usages but at much higher speeds and lower fees
- Is borderless
- Exciting new products possible! - Flash Loans

DeFi Ecosystem



DeFi - Is it all the rage?

Smart Contracts

- Do you trust bankers or developers?
- Many exploits - draining LPs
- Can't sue a smart contract - Luna/Terra collapse

Industry

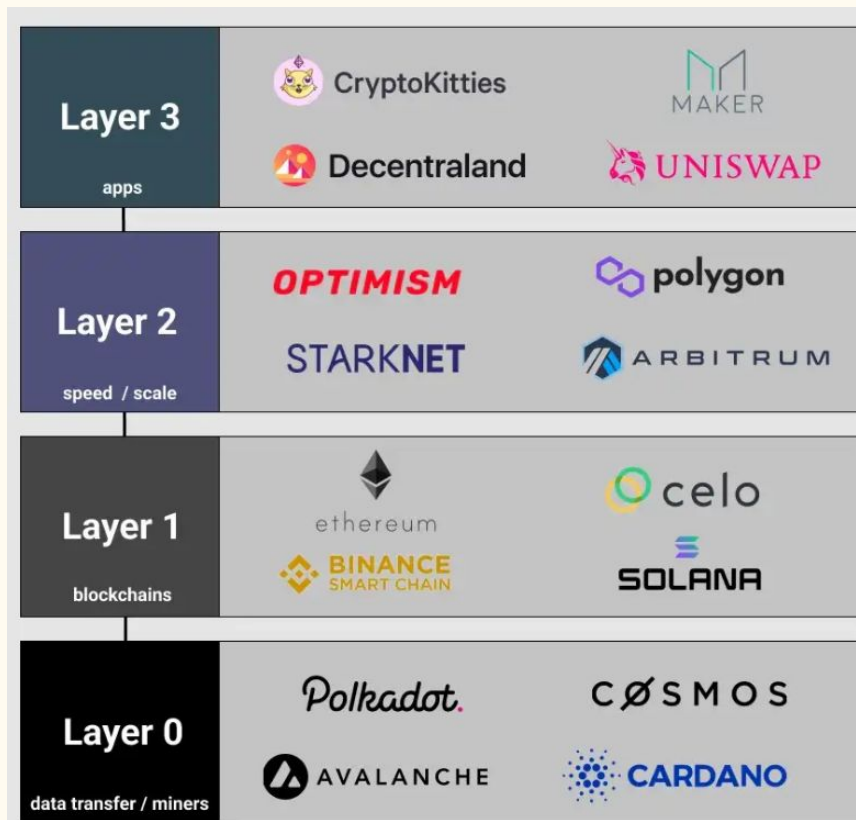
- Many scams and products
- Regulation has not yet caught up
- Is decentralisation an illusion?

Financial Performance

- Regular price/fee shocks
- How do you value hype?

DeFi Stack & Evaluation

ATTENTION: Don't confuse the stacks



Applications

dApps, Games, Metaverse, DEX

Scaling

Rollups, ZK

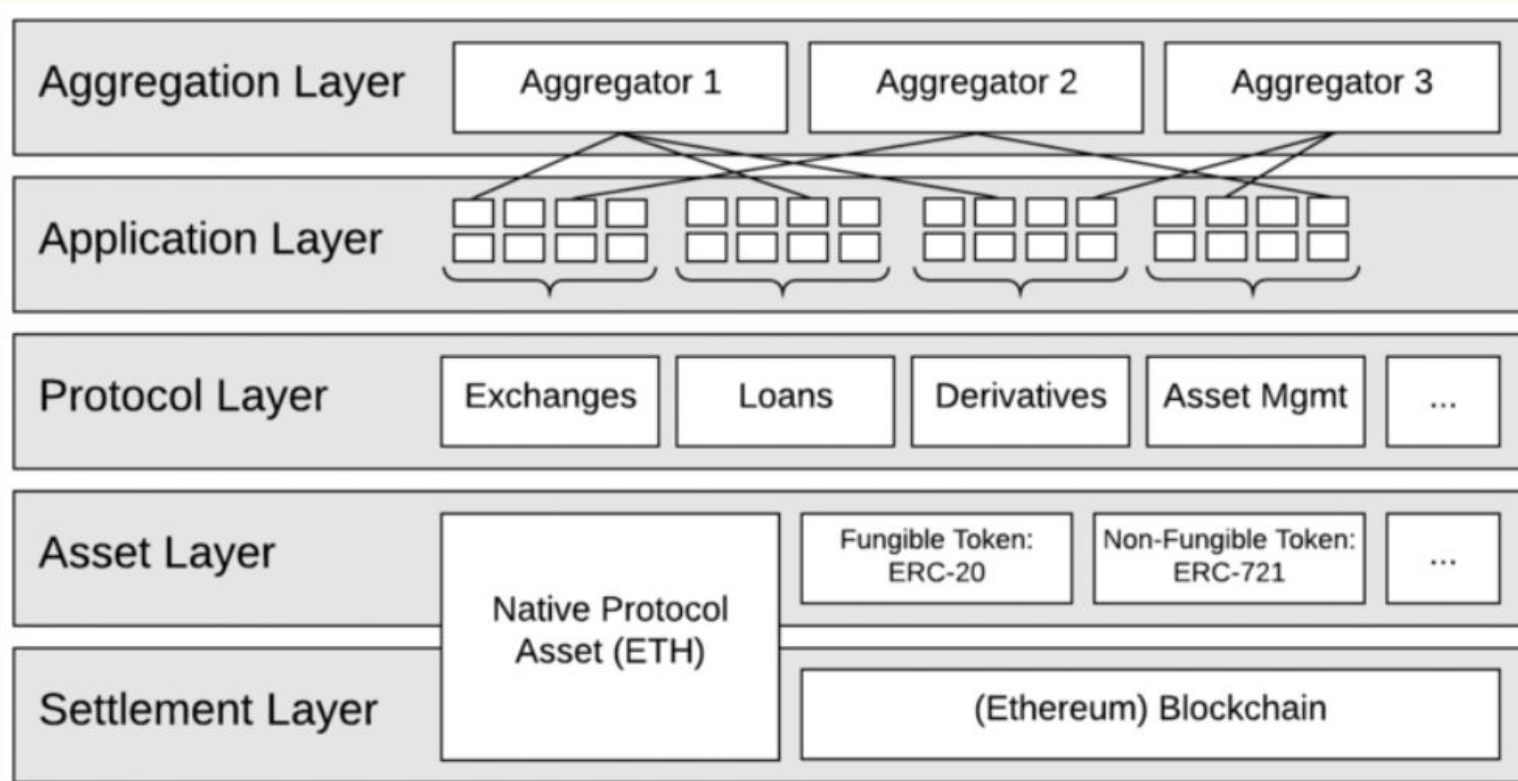
The Chain

Set chain characteristics, Foundation + community

Create the blockchain

Consensus Mechanisms, Cryptography, Interoperability, etc.

The DeFi Stack



Protocol Evaluation Metrics

Financial Metrics

Total Value Locked (TVL) - Amount of tokens deposited in protocol x token price

Annual Percentage Yield (APY) - For exchanges. Interest earned

Network Value to Transactions (NVT) - Market Cap : Transaction Volume

Unique Addresses - these represent wallet addresses and smart contracts.

Community Metrics

Daily / Monthly Active Users - performed a trade, deposit, some action

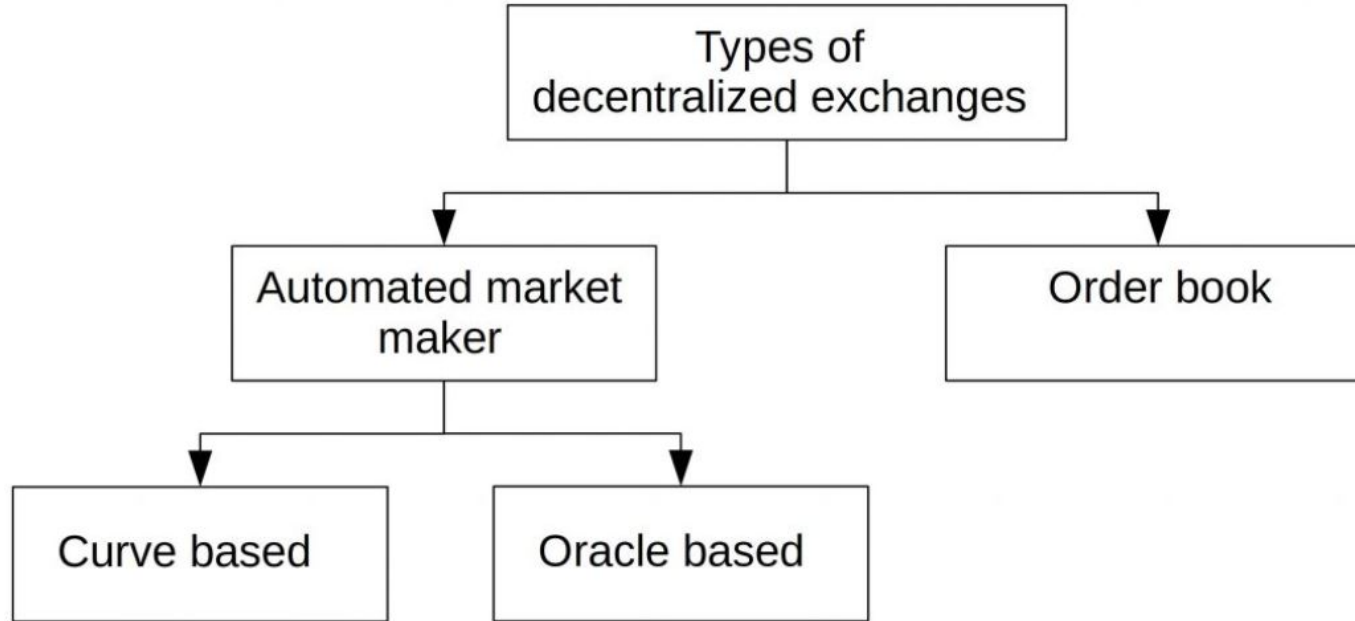
Active Developers - 10 commits every month (Electrical Capital)

Metcalf's Law - value of network = users in network squared

Exchanges and Models

TradFi & current ecosystem

Exchange Models



Order book Model - Binance

Price(BUSD)	Amount(BTC)	Total
23112.21	0.00200	46.22442
23112.20	0.15143	3,499.88045
23112.15	0.08000	1,848.97200
23112.13	0.00100	23.11213
23111.88	0.02164	500.14108
23,114.63↑ \$23,107.70		More
23113.55	0.00100	23.11355
23113.48	0.00400	92.45392
23113.46	0.27971	6,465.06590
23113.44	0.08000	1,849.07520
23113.04	0.00400	92.45216

Sell Order / Ask / Market Maker

The lowest price that a seller is willing to sell at.

Price: 23112,21 BUSD for 1 BTC

Amount: 0,002 BTC available at this price

Total: Price x Amount = 46,22 BUSD can be sold at this price level.

Buy Order / Bid / Price Taker

The highest price that a buyer is willing to buy at.

Price: 23113,55 BUSD for 1 BTC

Amount: 0,001 BTC available at this price

Total: Price x Amount = 23,11 BUSD will be exchanged at this price level.

Order book depth - Binance



Bid Ask Spread - Binance



Exchange price - Binance



Order Types - Market Order

Buy or sell at the current market price.

1. The trade is executed immediately at the price set by the exchange.

2. The market price is usually slightly higher than average buyer bids and slightly lower than seller asks. Profit from the spread!

Pros:

Guaranteed order completion

Immediate result - no waiting

Cons:

Price not guaranteed.

Slippage - placed vs executed

Order Types - Limit Order

Set a limit on the buy and sell price. Order is either fulfilled or expires at a certain time frame.

1. Place an order with a set limit.
2. Order gets executed when market price reaches the limit
3. Due to how the market price is set, the exchange would have bought at when asks were lower and sell when bids are higher. Still profiting from the spread!

Pros:

Price control.

Cons:

A chance that the order is not fulfilled or partially fulfilled.

Loses on other exchange fees.

Order Types - Some Others

Market Protection

Stop-Limit Order - Slippage Protection

Place the limit order at the stop price

Example: BTC is trading at 10k. You place an order with limit 9k and stop 9.5k

Your order to sell at 9k will be placed when BTC is selling at 9.5k (decline!)

One cancels the Other (OCO) - Hedge your bets

As soon as one order is fulfilled, cancel the other

Example: BTC is trading at 10k. You place a buy order at 9k and a sell order at 11k.

If Bitcoin price rises, sell and cancel buy order.

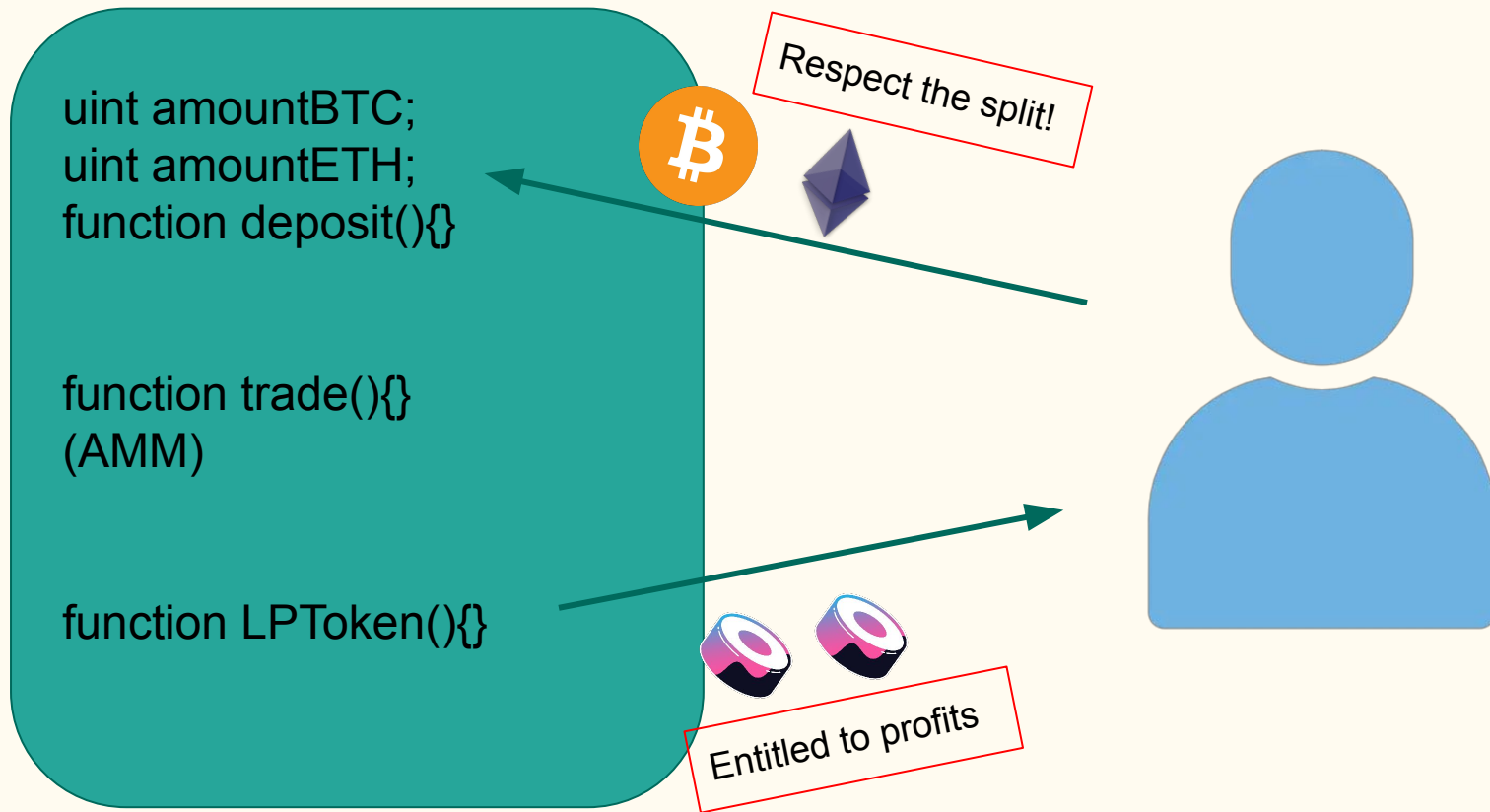
Time Protection

Good till Canceled - Keep this order open until I manually cancel

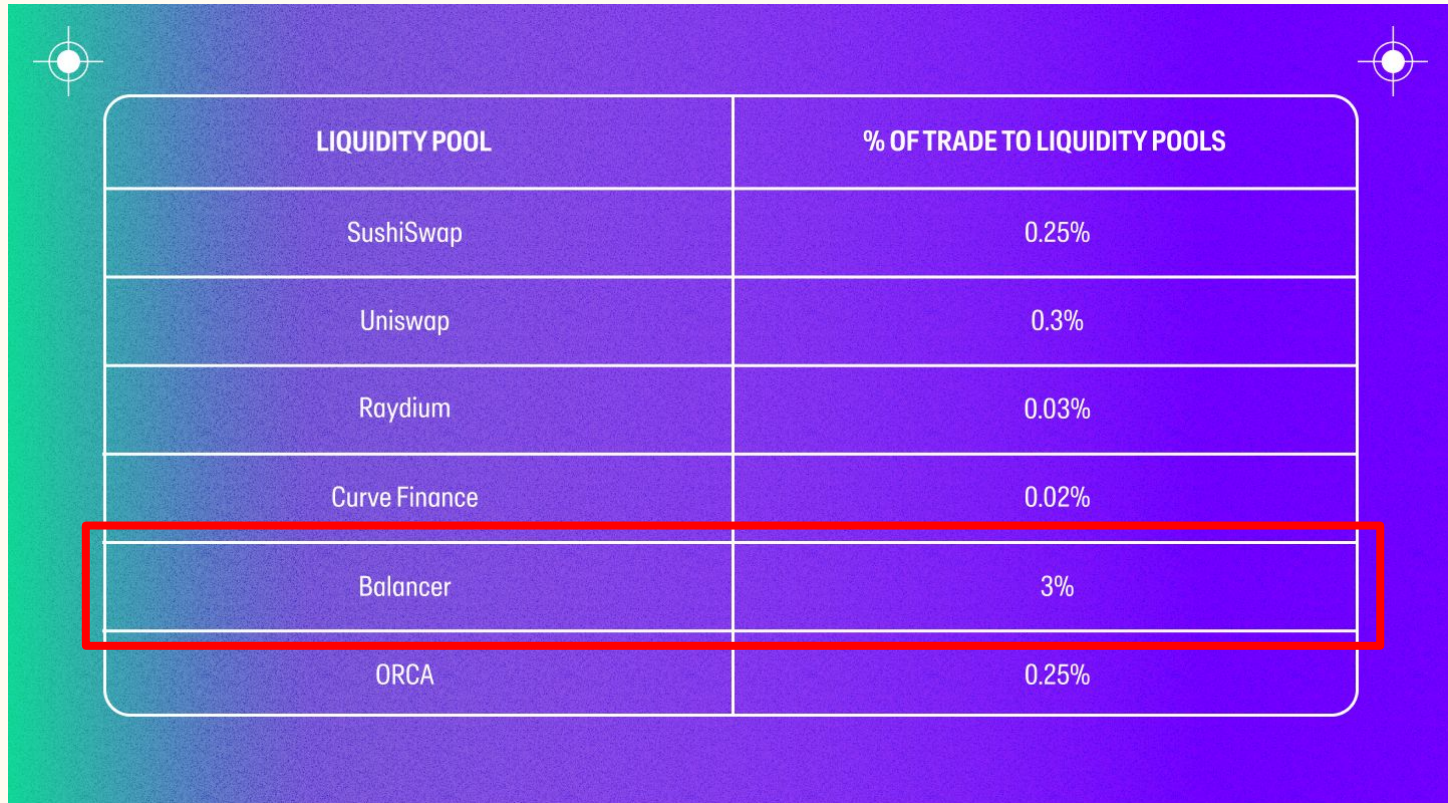
Immediate or Cancel - Immediately fulfill whatever is possible and cancel the rest

Fill or Kill - If full order not possible to fulfill, cancel everything

Onto DeFi - Liquidity Pools



Liquidity Pools - Fees

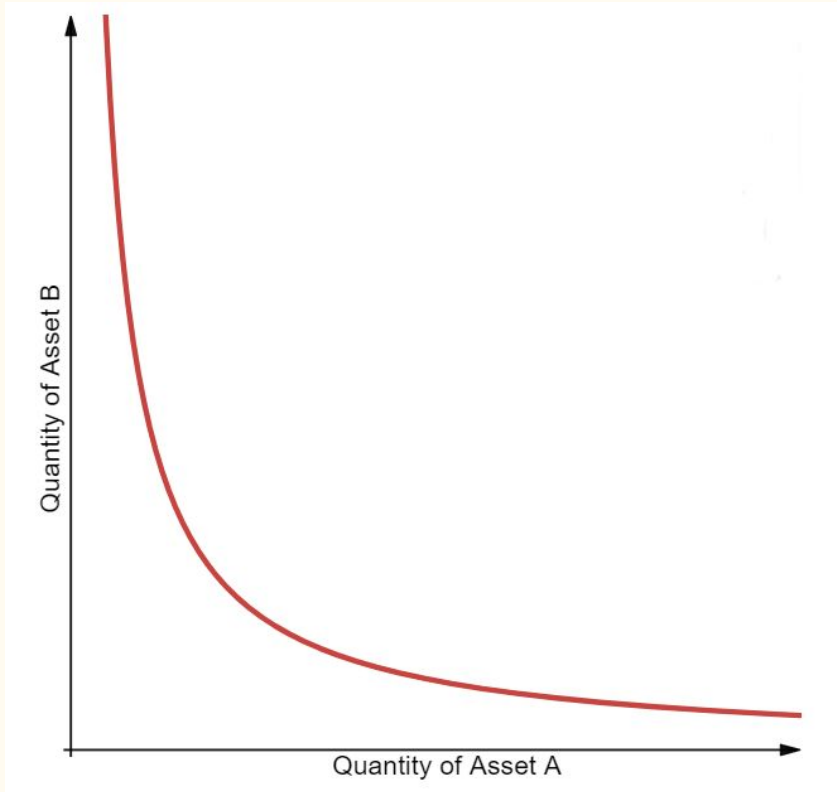


LIQUIDITY POOL	% OF TRADE TO LIQUIDITY POOLS
SushiSwap	0.25%
Uniswap	0.3%
Raydium	0.03%
Curve Finance	0.02%
Balancer	3%
ORCA	0.25%

AMM - Curve

Algorithmic price setting

Constant Product Model: CP-AMM - Uniswap



$$A * B = k$$

$$(Rx - \Delta x)(Ry + (1 - f)\Delta y) = k$$

CPAMM - Examples

The Liquidity Pool currently contains 1000 ETH and 10 BTC.

Question 1: What is the current product constant? What is the current exchange rate?

$$1000 \times 10 = 10\,000$$

$$100 \text{ ETH} : 1 \text{ BTC}$$

Question 2: I want to buy 1 BTC. How much ETH do I have to pay? What is the new exchange rate?

$$(1000 - x) \times (10 - 1) = 10000$$

$$1000 - x = 10000 / 9$$

$$111,11 \text{ ETH} : 1 \text{ BTC}$$

$$x = 111,11 \text{ ETH}$$

CPAMM - Examples Cont.

The Liquidity Pool currently contains 1000 ETH and 10 BTC.

Question 3: I want to buy 5 more BTC. What is the exchange ratio now?

$$(111,11 - x) \times (9-5) = 10000$$

$$111,11 - x = 10000 / 4$$

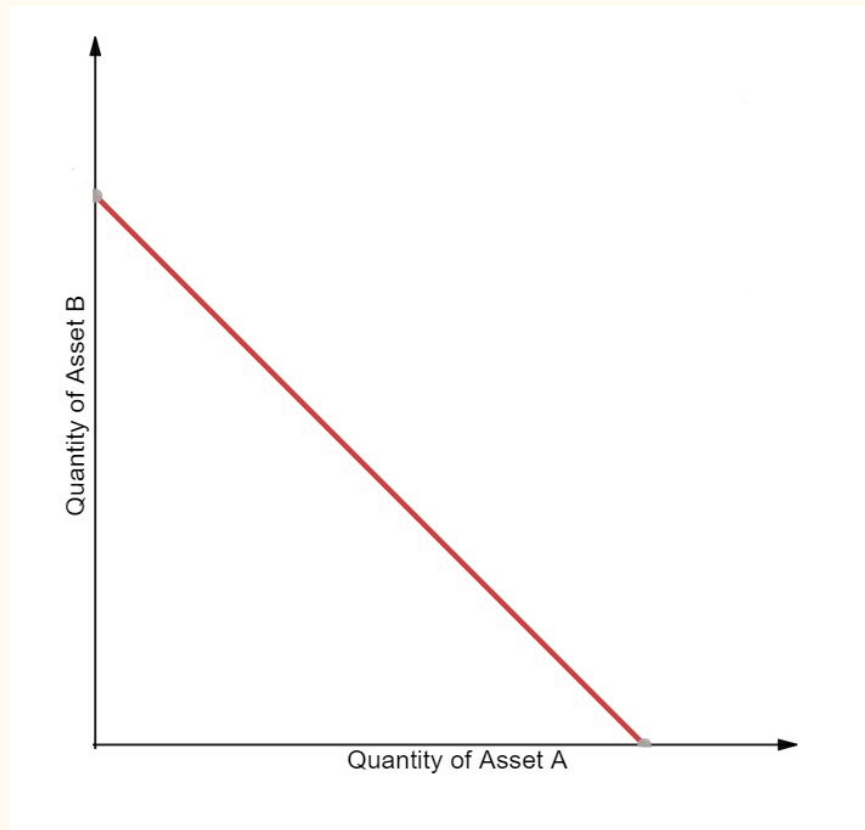
$$x = 2388,89 \text{ ETH}$$

$$2388,89 \text{ ETH} : 5 \text{ BTC}$$

$$477,78 \text{ ETH} : 1 \text{ BTC}$$

Towards the extremes, price volatility is exponential!

Constant Sum Model: CS-AMM



$$A + B = k$$

CSAMM - Examples

The Liquidity Pool currently contains 1000 USDT and 1000 USD.

Question 1: What is the current product constant? What is the current exchange rate?

$$1000 + 1000 = 2\,000$$

$$1\text{ USDT} : 1\text{ USD}$$

Question 2: I want to buy 100 USDT. How much USD do I have to pay? What is the new exchange rate?

$$(1000 - x) + (1000 - 100) = 2000$$

$$1000 - x = 2000 - 900$$

$$x = 100$$

$$1\text{ USDT} : 1\text{ USD}$$

CPAMM - Examples Cont.

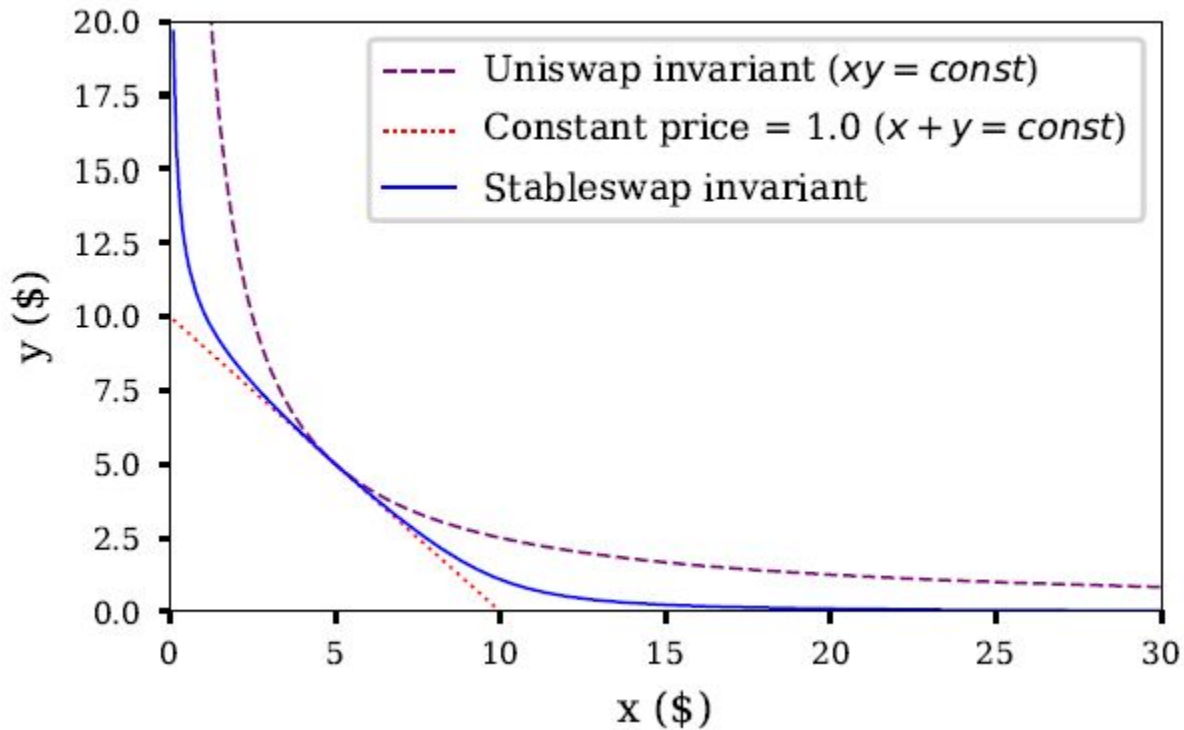
The Liquidity Pool currently contains 1000 ETH and 10 BTC.

Question 3: I want to buy the remaining 900 USDT. What happens?

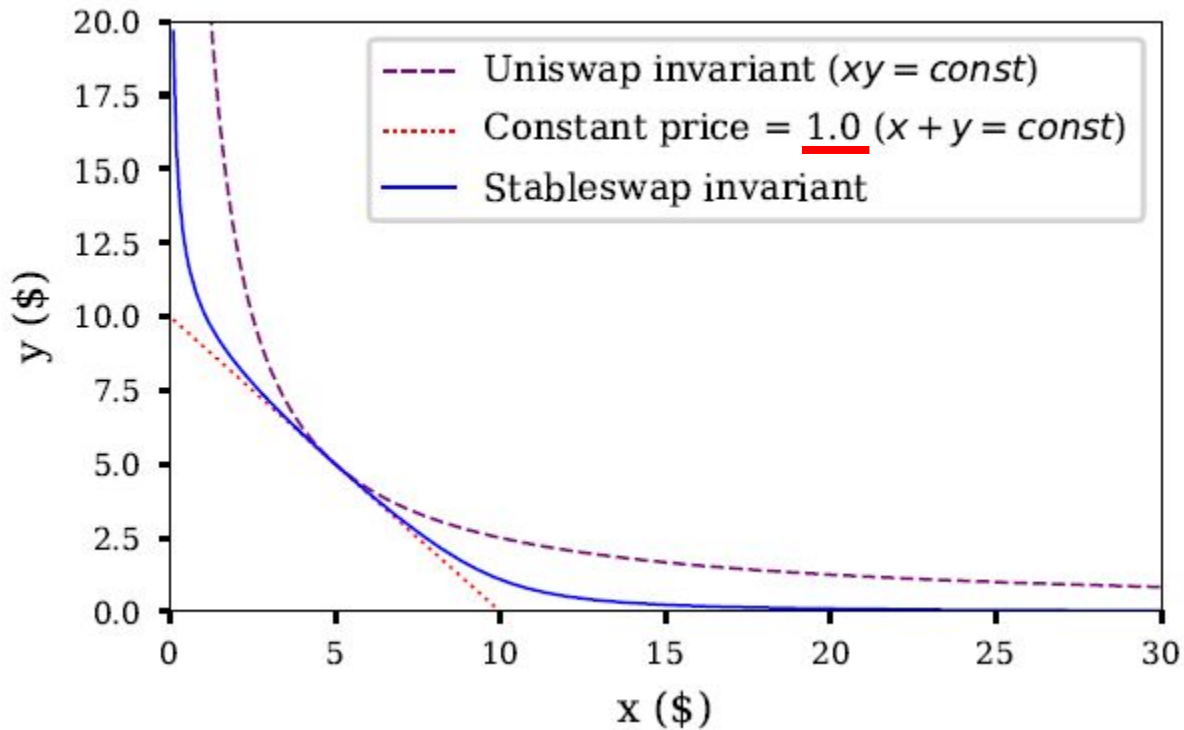
0 USDT remaining!!!

Price stable but liquidity danger!!!

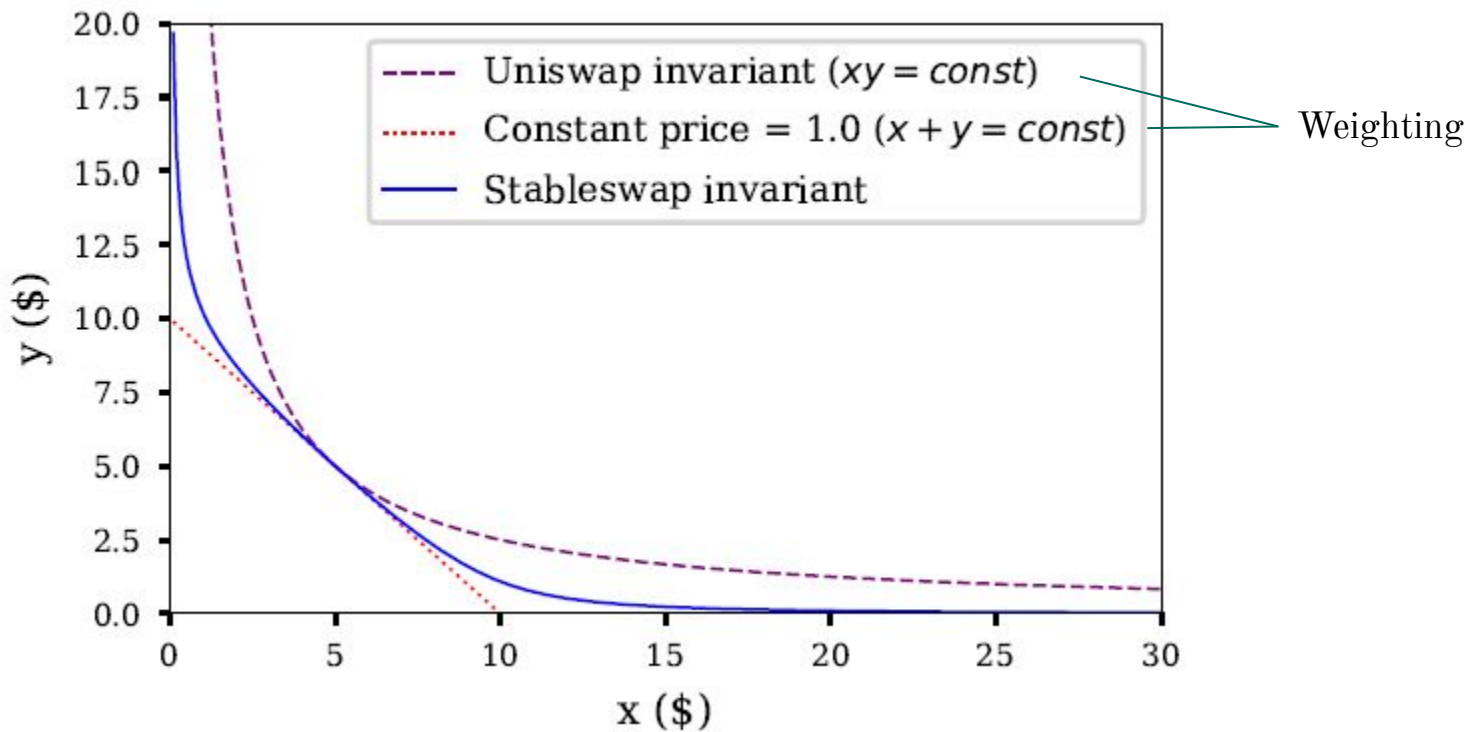
Hybrid CF-AMM - Curve Finance



We need data! - Break free from 1:1



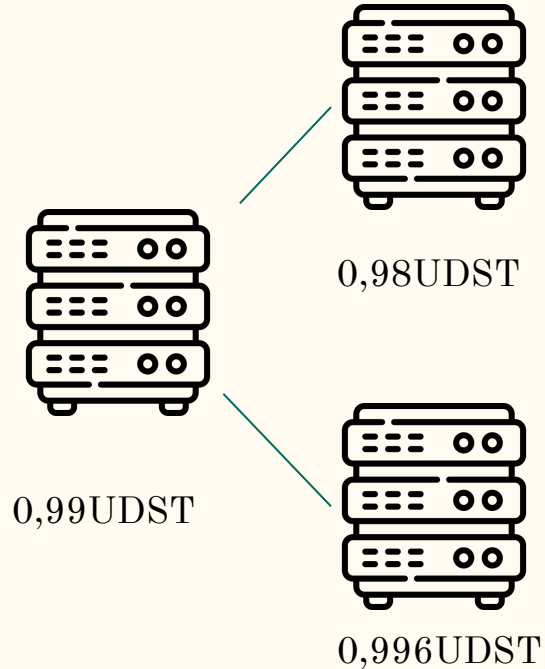
We need data! - Dynamic leverage



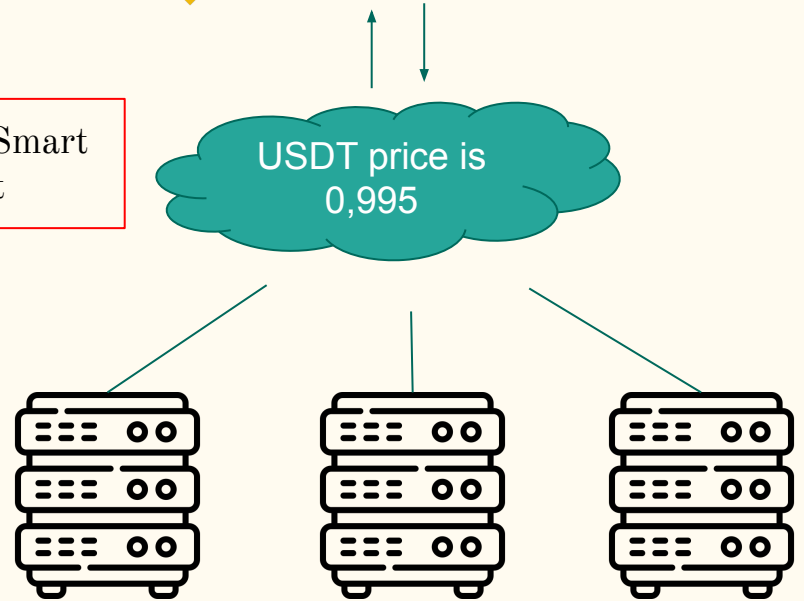
AMM -Oracles

Pulling external data onchain

Data Volatility and Consensus



Hybrid Smart Contract



What is the price?

Oracle Problem - Data Quality

Correctness

Authenticity - Is the data real. 18 degrees in winter?

Integrity - http network attacks. Did data get from A to B safely and unaltered?

Availability

Data must be available when requested. Must not hold up time sensitive executions.

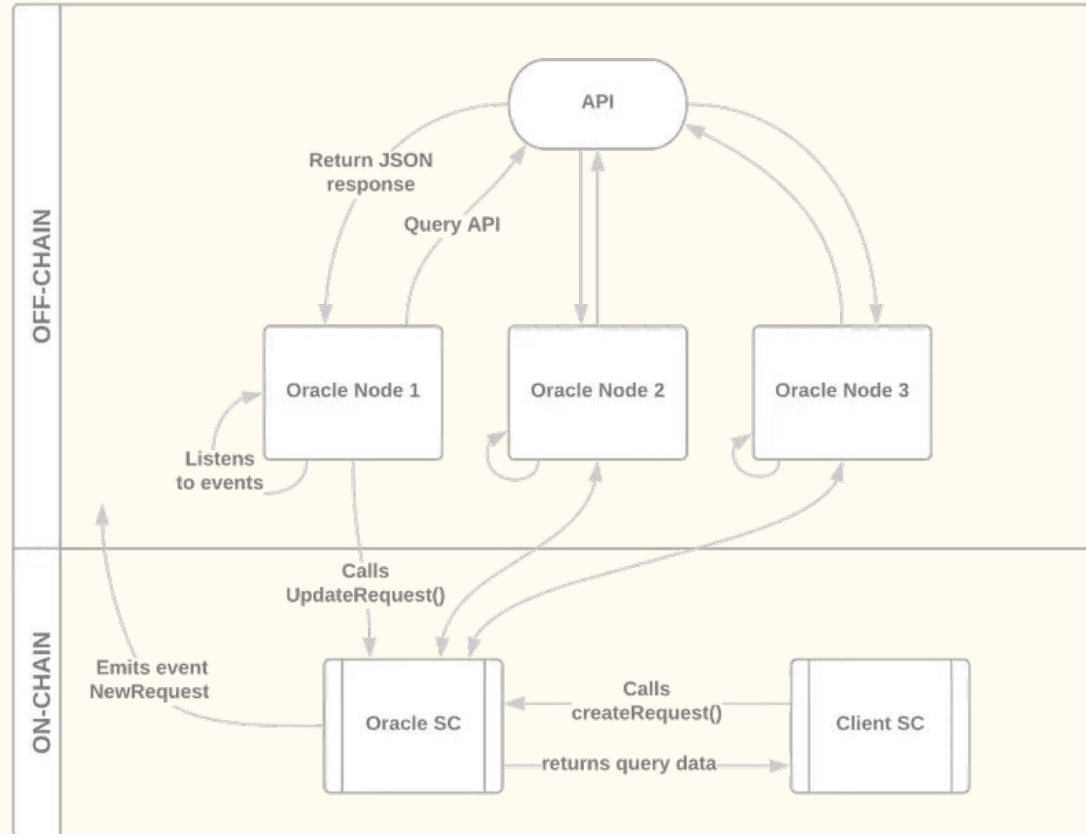
Incentive Compatibility

How to reward data providers so data quality increases

Attributability - Who supplied the data

Accountability - Appropriate reward and punishment

Oracle Architecture



Oracle Example - State

```
1  pragma solidity >=0.4.21 <0.6.0;  
2  
3  contract Oracle {  
4      Request[] requests; //list of requests made to the contract  
5      uint currentId = 0; //increasing request id  
6      uint minQuorum = 2; //minimum number of responses to receive before declaring final result  
7      uint totalOracleCount = 3; // Hardcoded oracle count  
8  }
```

Oracle Example - Requests

```
1  // defines a general api request
2  struct Request {
3      uint id;                                //request id
4      string urlToQuery;                      //API url
5      string attributeToFetch;                //json attribute (key) to retrieve in the response
6      string agreedValue;                    //value from key
7      mapping(uint => string) answers;        //answers provided by the oracles
8      mapping(address => uint) quorum;        //oracles which will query the answer (1=oracle hasn't voted,
9  }
```

Oracle Example - Result Collection

1. Check if the result comes from the list of "trusted" Oracles
2. Record the answer and mark the Oracle as "voted"
3. Loop through all the answers and compare
4. When an agreement is found, increment the Quorum Count.
5. Check when Quorum is reached
6. Update Request to the requesting smart smart

Oracle Example - Update Requests

```
1  //triggered when there's a consensus on the final result
2  event UpdatedRequest (
3      uint id,
4      string urlToQuery,
5      string attributeToFetch,
6      string agreedValue
7  );
```

Oracle Architecture Types

Request-Receive

Model in previous example

A request is made and data is gathered from sources

Suitable for very large datasets.
Users only require a small portion from the set.

Publish-Subscribe

Deals with constant data streams like pricing data.

An Oracle subscribes to a data stream, gets notified of updates in frequent intervals

Client contracts, will listen for updates like an event listener in Javascript

Data Providers

Centralized

All off chain nodes feeds into a central oracle that makes a final decision

Suitable for proprietary data sets (Statista, World Bank, Forbes, Stiftung Warentest)

Pros:

No consensus needed - fast decision

Cons:

No checks resulting in low correctness

Low data diversity - DOS attacks possible

Decentralized

Multiple onchain Oracles come to a consensus

Pros:

Authenticity proofs (TLS proof, Trusted Execution Environments)

Voting or staking on the accuracy of data - Sybil attacks!

High availability - if one Oracle goes down, many others

Incentivized - on chain oracles have reputation scores