Lecture 5

Intro to DeFi

DeFi Overview

TradFi & current ecosystem

TradFi - Traditional Finance

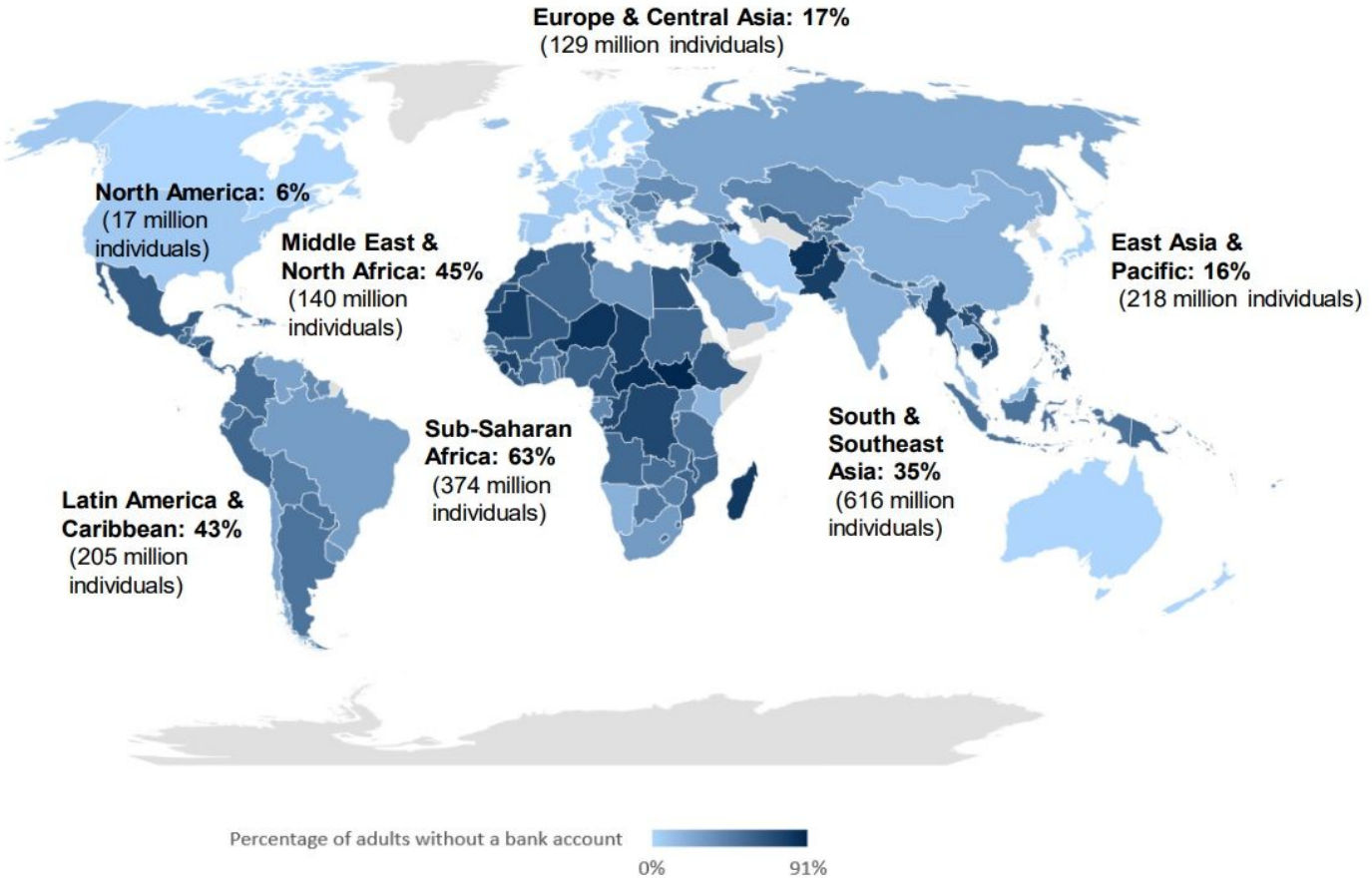
A **centralized** trusted authority that enables trustless transactions.

- Provides assurance of value
- Offers financial products
- Enforces compliance, fight 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 groups
- 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

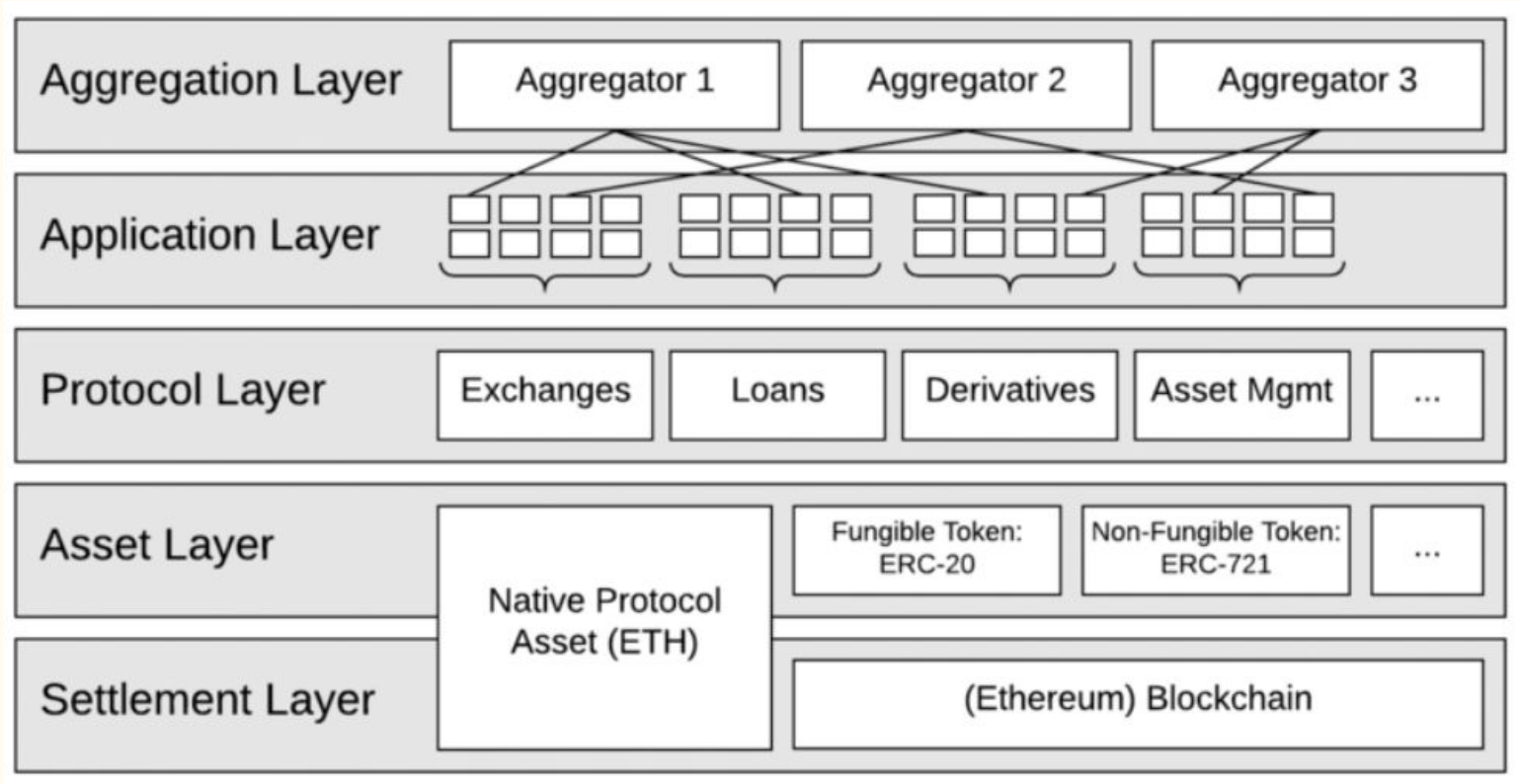


Decentralized Finance - What is DeFi?

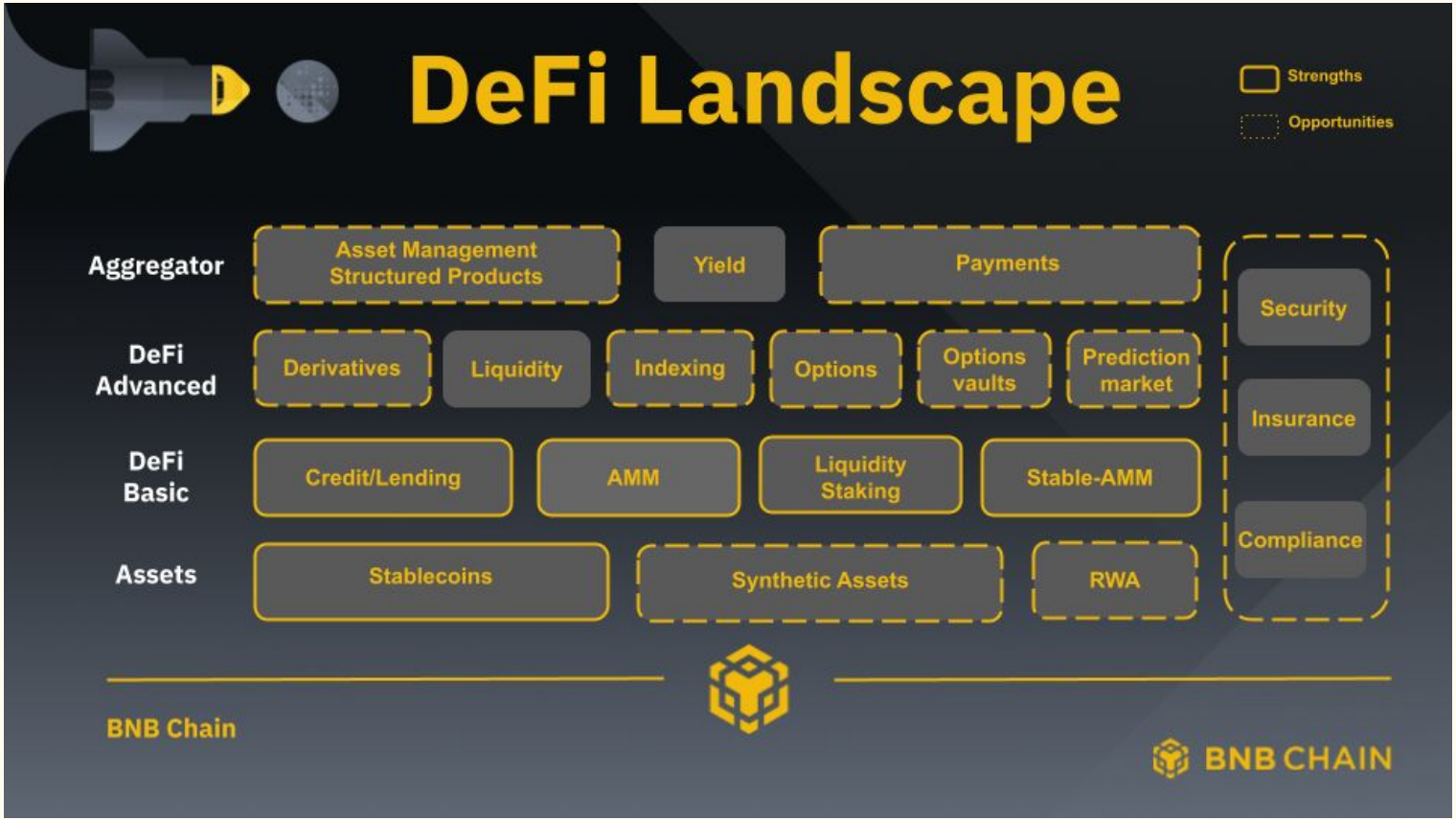
A group of financial software products and services that:

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

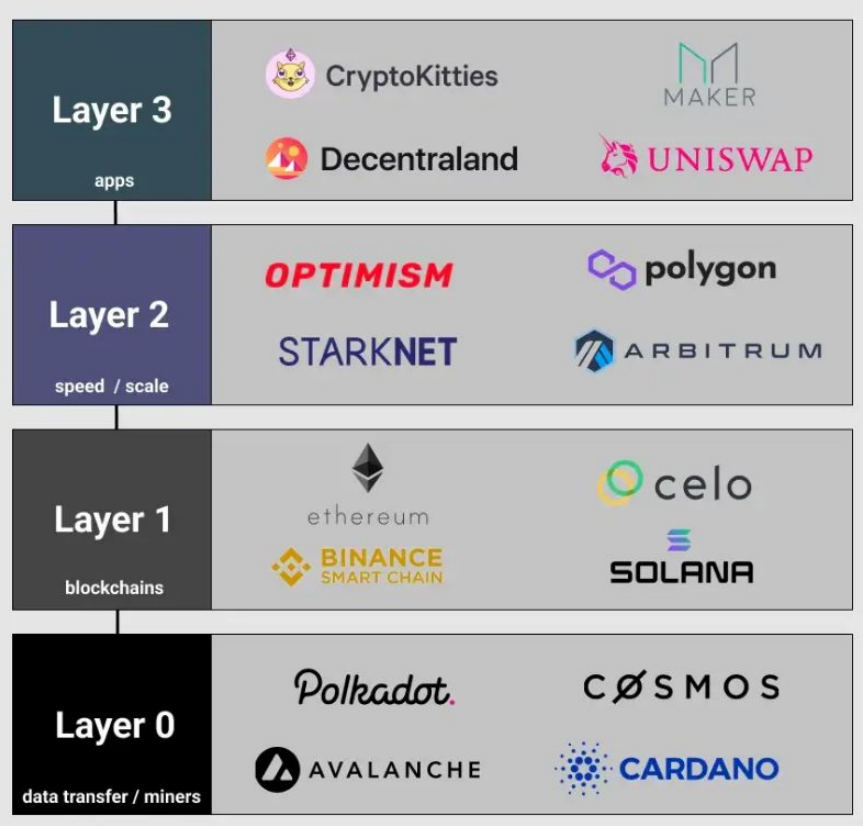
The DeFi Stack



DeFi Services on Stack



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.

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 valuate hype?

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

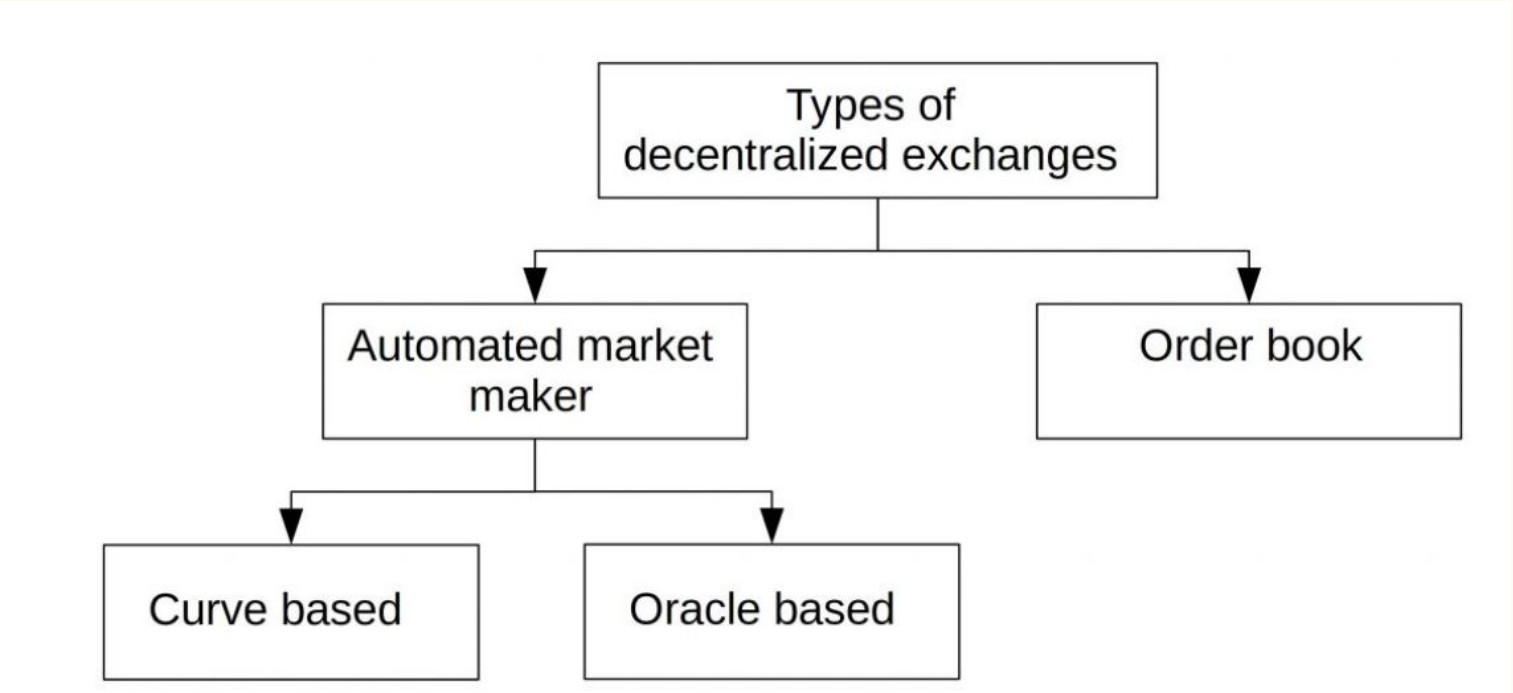
Active Developers - 10 commits every month (Electrical Capital)

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

Exchanges and Models

TradFi & current ecosystem

Exchange Models

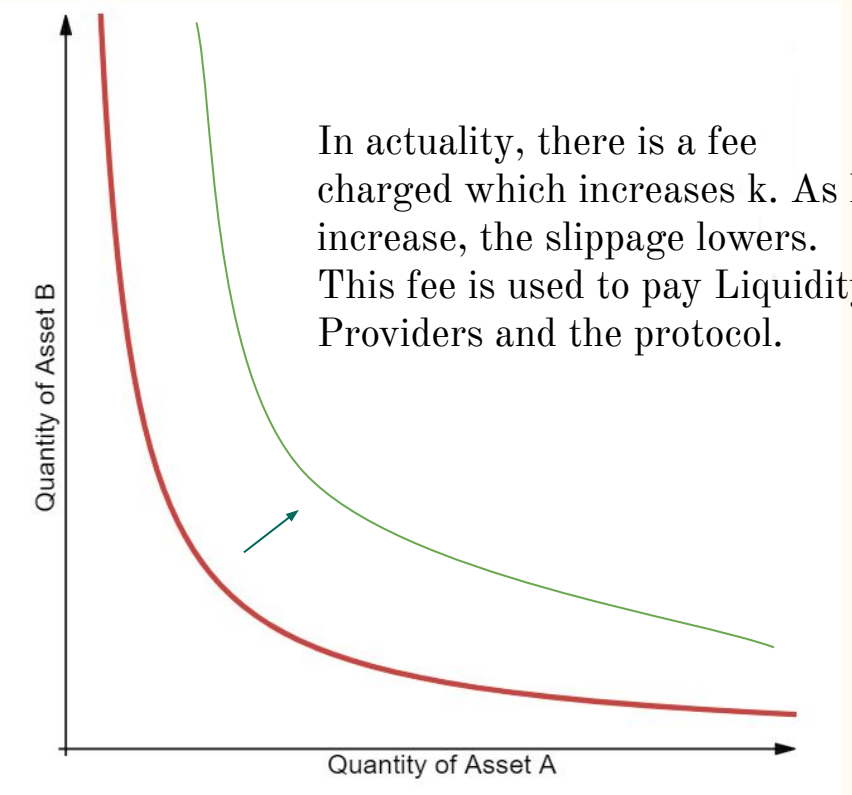


https://blockchain-academy.hs-mittweida.de/courses/blockchain-introduction-technical-beginner-to-intermediate/lessons/lesson-26-decentralized-exchanges/topic/introduction-to-decentralized-exchanges/

AMM - Curve

Algorithmic price setting

Constant Product Model: CP-AMM - Uniswap



In actuality, there is a fee charged which increases k. As k increase, the slippage lowers. This fee is used to pay Liquidity Providers and the protocol.

$$A * B = k$$

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 x 10 = 10 000

100 ETH : 1BTC

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

(1000 - x) * (10 - 1) = 10000

1000 - x = 10000 / 9

x = 111,11 ETH

111,11 ETH : 1BTC

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) x (9-5) = 10000

111,11 - x = 10000 / 4

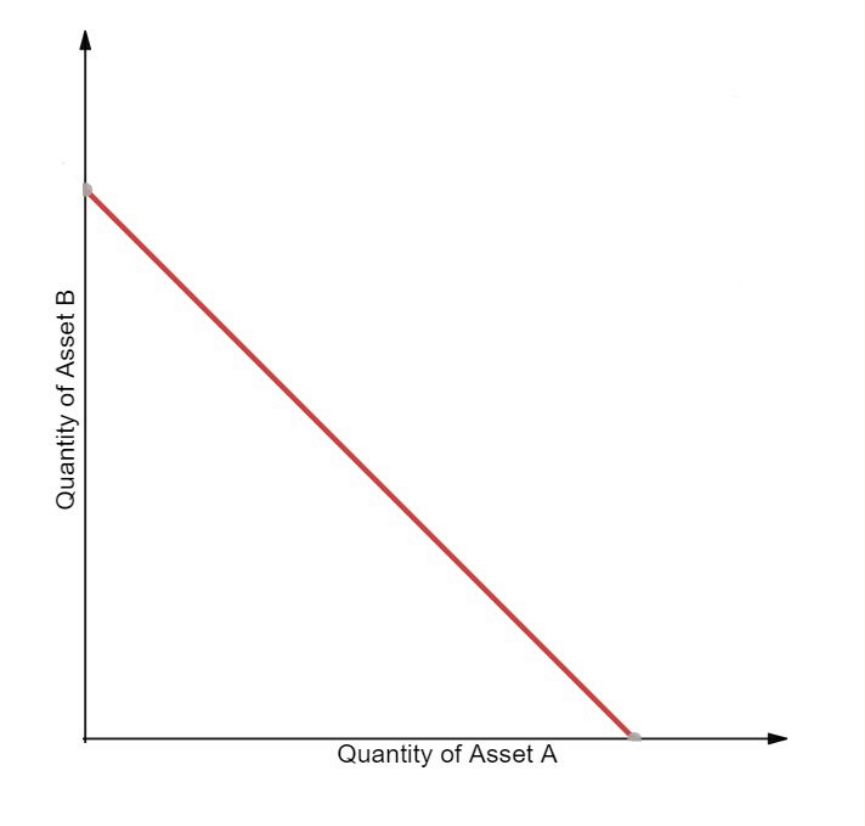
x = 2388,89 ETH

2388,89 ETH : 5 BTC

477,78 ETH : 1BTC

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 USDT : 1USD

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 USDT : 1USD

CPAMM - Examples Cont.

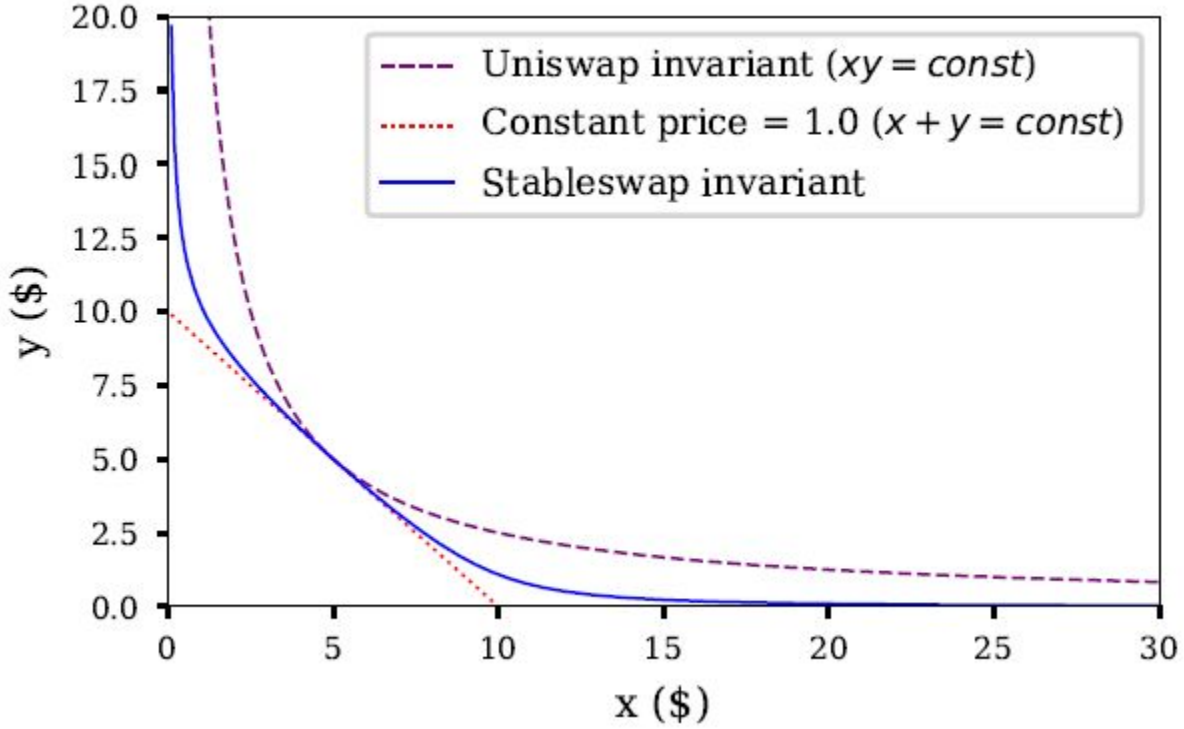
The Liquidity Pool currently contains 1000 ETH and 10 BTC.

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

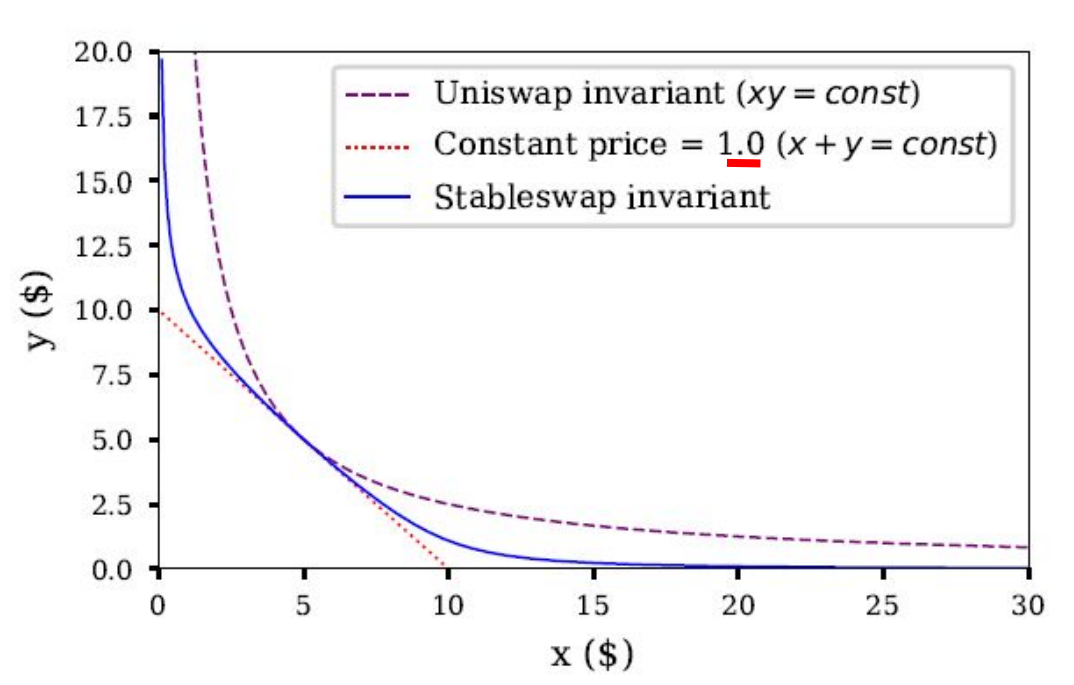
0 USDT remaining!!!

Price stable but liquidity danger!!!

Hybrid CF-AMM - Curve Finance

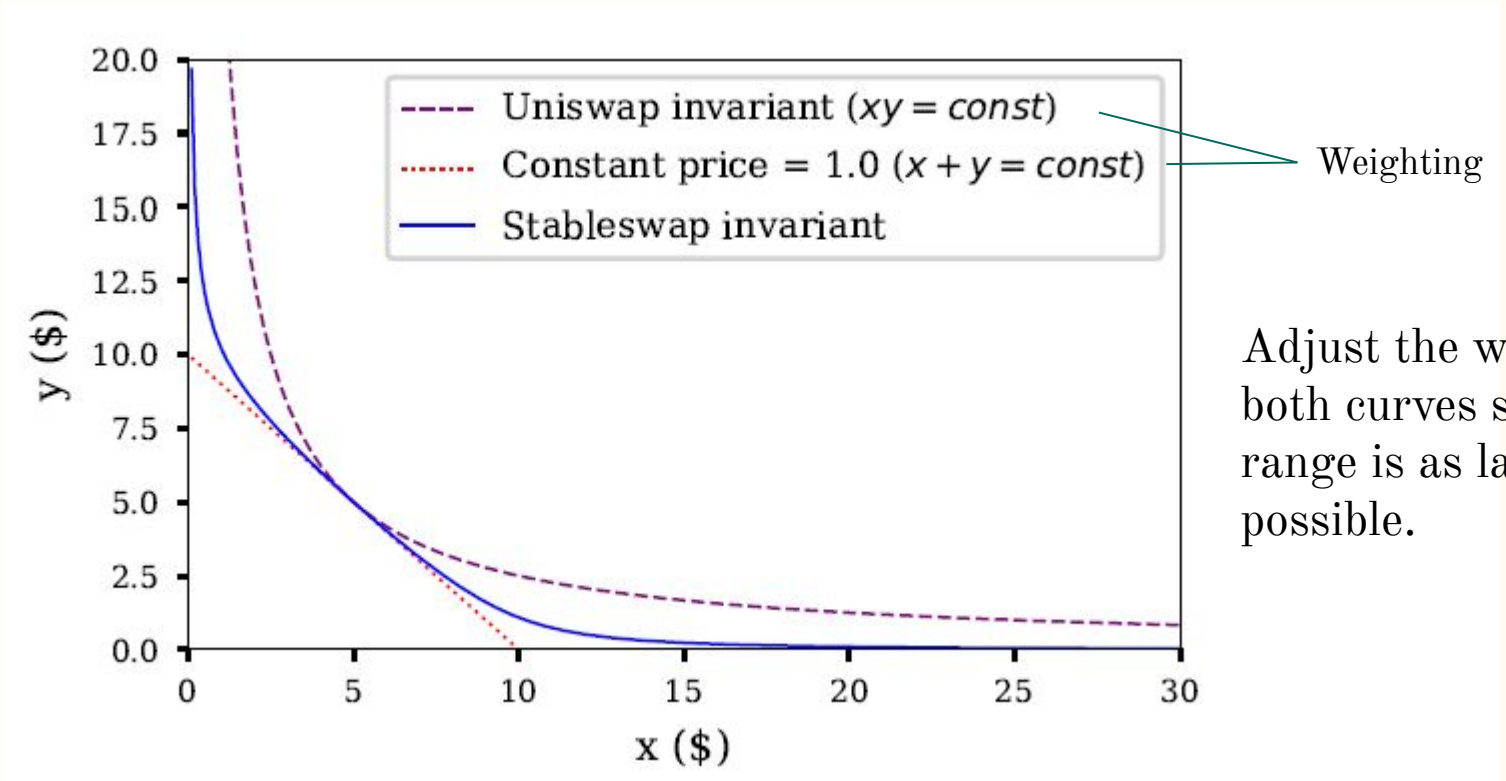


We need data! - Break free from 1:1



Changing the constant factor flattens the curve (ends become straight like the line), decreasing price volatility and slippage.

We need data! - Dynamic leverage



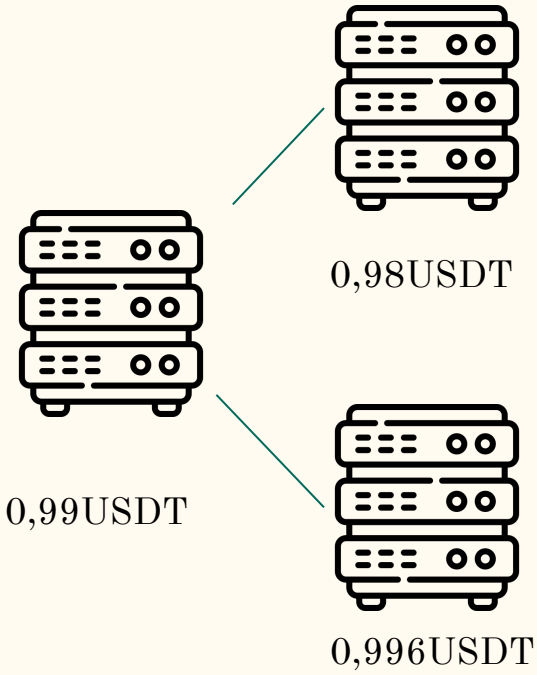
Weighting

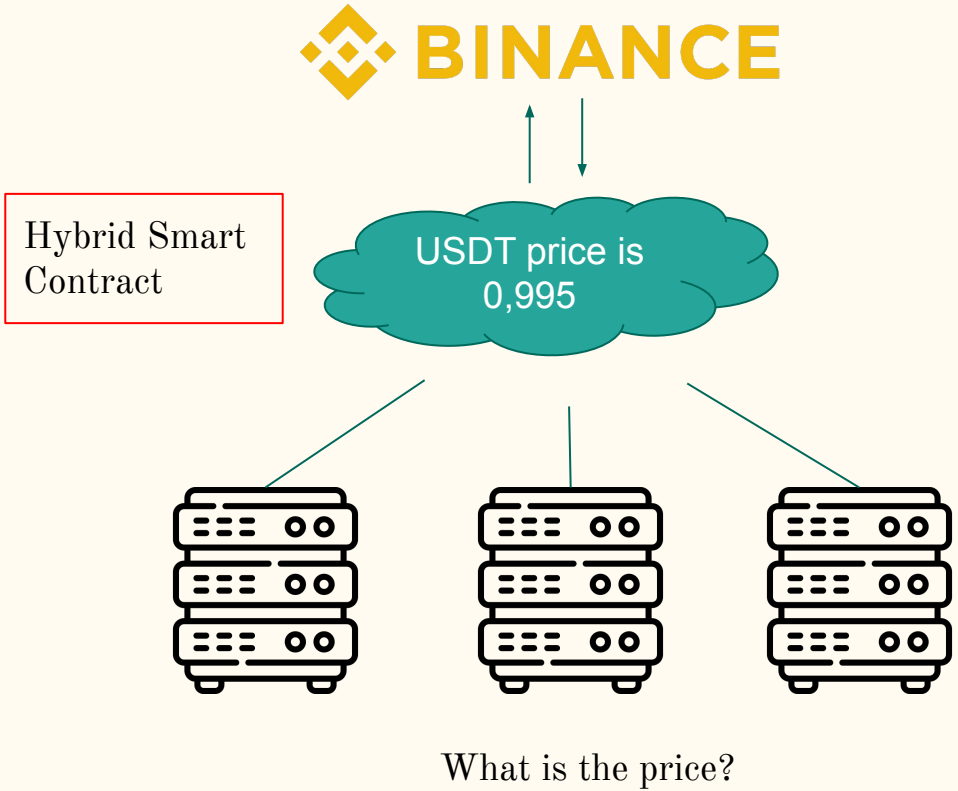
Adjust the weighting of both curves so the stable range is as large as possible.

AMM -Oracles

Pulling external data onchain

Data Volatility and Consensus





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

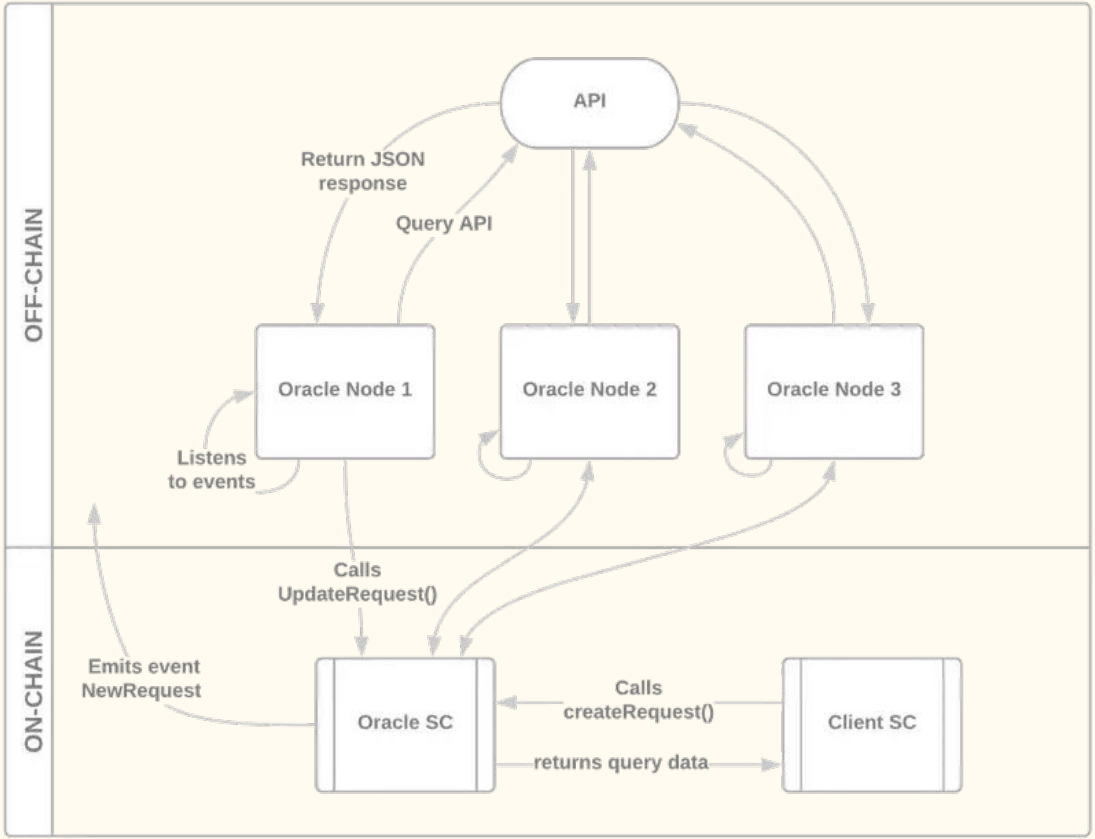
How to reward data providers so data quality increases

Attributability - Who supplied the data

Accountability - Appropriate reward and punishment

Oracle Architecture

Emitted events are shown on chain, and are detected by data indexers (like GraphQL) interfacing with Validators



UpdateRequest is a SC call going through EVM nodes

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) anwers;   //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 - Client Agreements

Request-Receive
Model in previous example A request is made, fee payment and data is gathered from sources Suitable for once off, non time critical requests. Users only require a small unit of data.

Publish-Subscribe
Deals with constant data streams like pricing data. A client subscribes to an Oracle data stream, gets notified of updates in frequent intervals. Fee payment for a full interval. Client contracts, will listen for updates like an event listener in Javascript

Data Providers - Offchain Nodes

Centralized	Decentralized
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	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

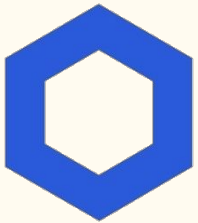
https://www.youtube.com/watch?v=6AzhClN240M

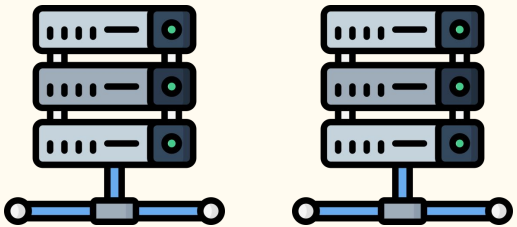
Tokenomics review! - Utility Tokens

A token which provides access to specific product or service on a blockchain

- Receives special/limited edition NFTs
- Payment for specific services
  - Filecoin - storage
  - Chainlink - data
- Ecosystem Perks
  - BNB holders enjoy 25% reduced fees
  - Web2 equivalent - Vente Privee

LINK - Tokenomics Analysis Example





Smart Contract pay Oracles LINK (now ETH also possible) for this information retrieval and verification. Everyone needs data, huge addressable market.

A Service Level Agreement is created:

1. Oracle reputation (In decentralized Oracles)
2. Order Matching - Nodes must bid to fulfill this data request. LINK used
3. Oracle contract - collects all bids and computes final results from all oracles

Penalties can be charged in LINK for fake / low quality data

Nodes are integration point with blockchain and traditional data sources

Use LINK tokens to vote which nodes can supply information to the Oracle - data reliability and verification.

Receive LINK as a reward for data delivered to an Oracle

RECAP - Blockchain Project Evaluation

- Tokenomics
  - What does the token lifecycle look like?
  - Are incentive mechanisms designed in accordance to its lifecycle?
  - Are incentives motivating **good user and community behaviour** and s**ecure and honest infrastructure**?
- Financial Metrics
  - TVL, Market Cap, Trading Volume, Wallet Addresses
  - Use aggregators, prediction models, contract audit reports
  - Financial / Risk modelling?
- Community Metrics
  - Github activities
  - Forum and DAO liveliness
  - Ambassadors

Lending and Borrowing

TradFi & current ecosystem

TradFi - How to take out a loan

1. Capital deposit between 10% - 15%.
    a. Immigrant discrimination in Germany? 40% (avg 20%).
2. Ask bank for a loan
    a. Type of work (CDI/CDD)
    b. Current fixed assets / debts
    c. Health state and insurances
    d. Histories - US Credit Scores vs German Schufa
3. Bank provides loan with interest rate and duration.
4. If you don't pay:
    a. Debt Collection
    b. Legal actions
    c. Repossession

DeFi Lending- Overcollaterization due to anonymity

Loan to Value ratio (LTV) - The value of the collateral compared to the loan. Commonly 100:150. This is also known as Collateralization Ratio (CR)

Liquidation Threshold - The point at which a loan is deemed to be under collateralized and actions are triggered in the smart contract.

LTV/LT - An example

We take out a loan of 1000 DAI. Simply assume DAI is a perfect stablecoin pegged at $1 always. We put in some ETH as collateral. Currently 1 ETH is worth $100.

Question 1 - How much ETH do we need to deposit for an LTV of 150%?

Solution 1 - We require $1500 worth of ETH. 1500/100 = 15 ETH

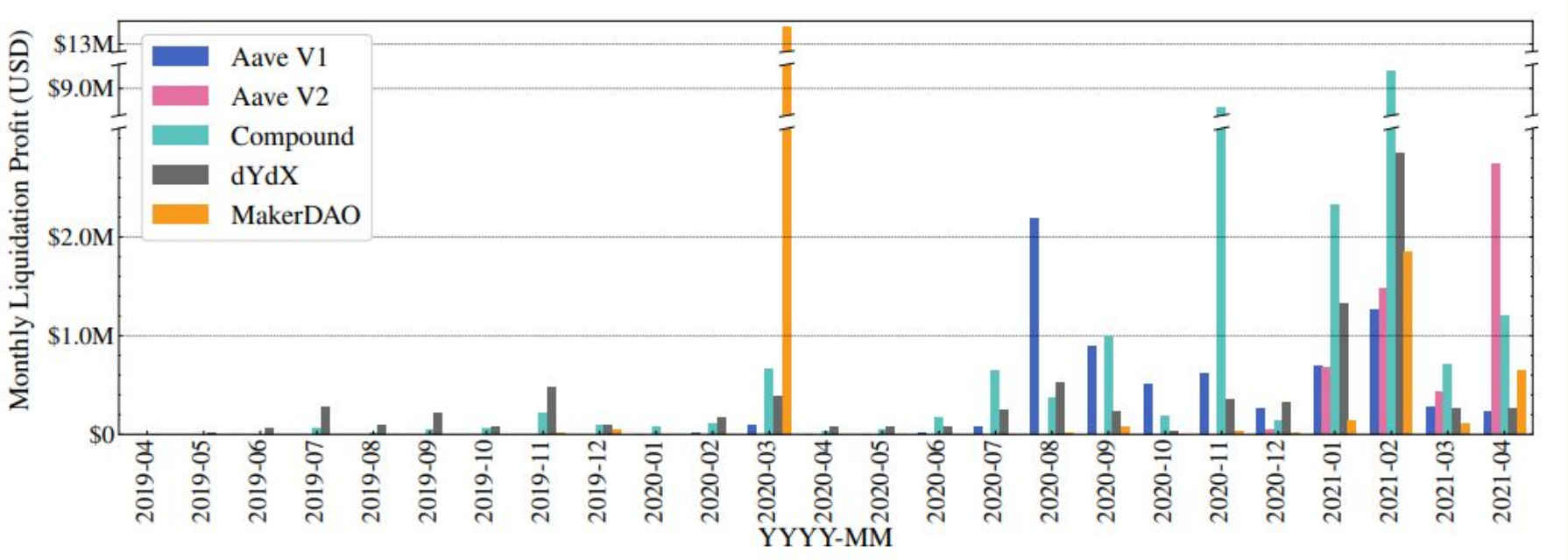
Question 2 - ETH prices drop by 10%. What is our new LTV?

Solution 2 - New price = $90. 90 x 15 = $1350. 1350/1000 = 135%

Question 3 - **The liquidation threshold is set at 110%, how much can the price of ETH drop?**

Solution 3 - LT = $1100 of ETH. We put in 15 ETH. 1100/15 = 73.33. About 27% drop in price.

What happens at LT? - Liquidations and profit



Occasionally, there can be massive bad debts and insolvency due to gas prices and liquidation mechanisms (remember gas and speed testing!)

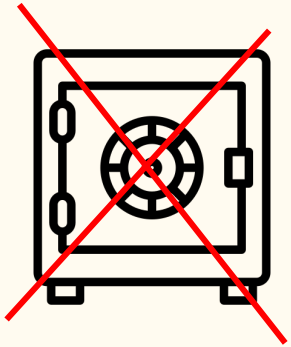
DeFi Case Study

MakerDAO

CDP and Stability Fees - Ideal Scenario

- DAI has a Liquidation Threshold of 150%.
- Users open a Collateralized Debt Position (CDP) with slightly higher collateralization ratio (avg 164%)
  - DAI Savings rate (demand generation mechanism)
  - Use DAI to buy more ETH for arbitrage
- Gets charged a **Stability Fee** - essentially the interest rate
  - Stability Fees paid upon repayment of DAI
  - Continuously calculated on DAI debt
  - Changes according to DAI mint and burn rates (supply control mechanism)
  - This is known as a soft pegging mechanism to control risk
- Repayment means a certain amount of DAI is burnt out of existence. If all goes well, DAI is minted and burned and some extra profit in the form of stability fees is harvested.

Liquidations - Debt is profitable



Vault is under LT ETH Collateral auctioned off for DAI + liquidation penalty

Auction recovered full DAI vault + Liquidation Penalty

Lender gets remaining ETH back, SC gets penalty fee.

Auction recovered full DAI vault but not enough Liquidation Penalty

Lender gets nothing, SC gets all remaining value

Auction not able to recover full DAI amount. Vault bankrupt!

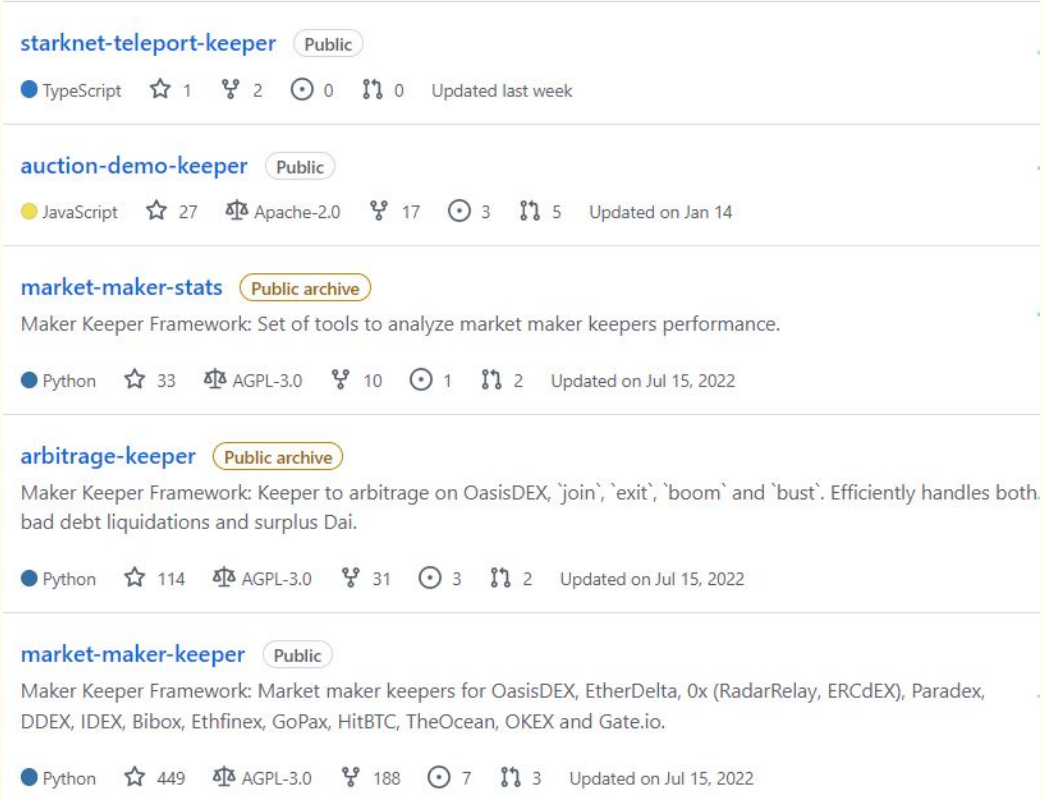
MKR token to the rescue. Mint MKR tokens to stabilize DAI

MKR not able to stabilize DAI. System Failure!!!

MakerDAO Keepers - stabilize the DAI peg

Smart Contract bots which continuously monitors for changes:

1. **CDP keeper** - manage open loans, set stability fees.
2. **Bite Keeper** - Monitor CDPs which falls under LT.
3. **Arbitrage keeper** - scans various DEXes for arbitrage opportunities
4. **Market Making Keeper** - market making activities for DAI on DEXes

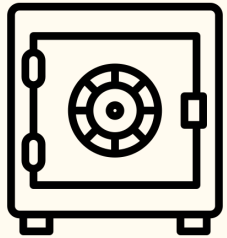


Maintaining System Stability -Auction Mechanisms

Collateral Auction

- Takes ETH available and starts auctioning for DAI.
- Bid Duration - Auction ends when no new bids within timeframe
- Auction Duration - Auction ends regardless

Vault is "bitten"



Stability Fee in DAI

Maker Buffer

DAI

Surplus Auction
Triggered when total DAI and debt reaches a certain ratio.

DAI auctioned for MKR.

Received MKR gets burnt. Decrease supply

Buffer attempts to absorb bad debt

Reverse collateral Auction
After initial interest and DAI availability, converts to auctions for purchasing DAI at decreasing ETH amounts

Debt Auction
No enough DAI was raised in the Collateral Auction or Reverse Collateral Auction.
Not enough DAI in buffer. System in in net debt state. Debt limit set through a vote.
Amounts of DAI for decreasing amout of MKR tokens.
MKR tokens gets minted. Supply increase.

Black Thursday

Irrationality overwhelms Technology