Tutorial 4:
Borrowing & Lending

CSCD71: Blockchains & Decentralized Applications

Nikhil Lakhwani, Oct 27 2023.

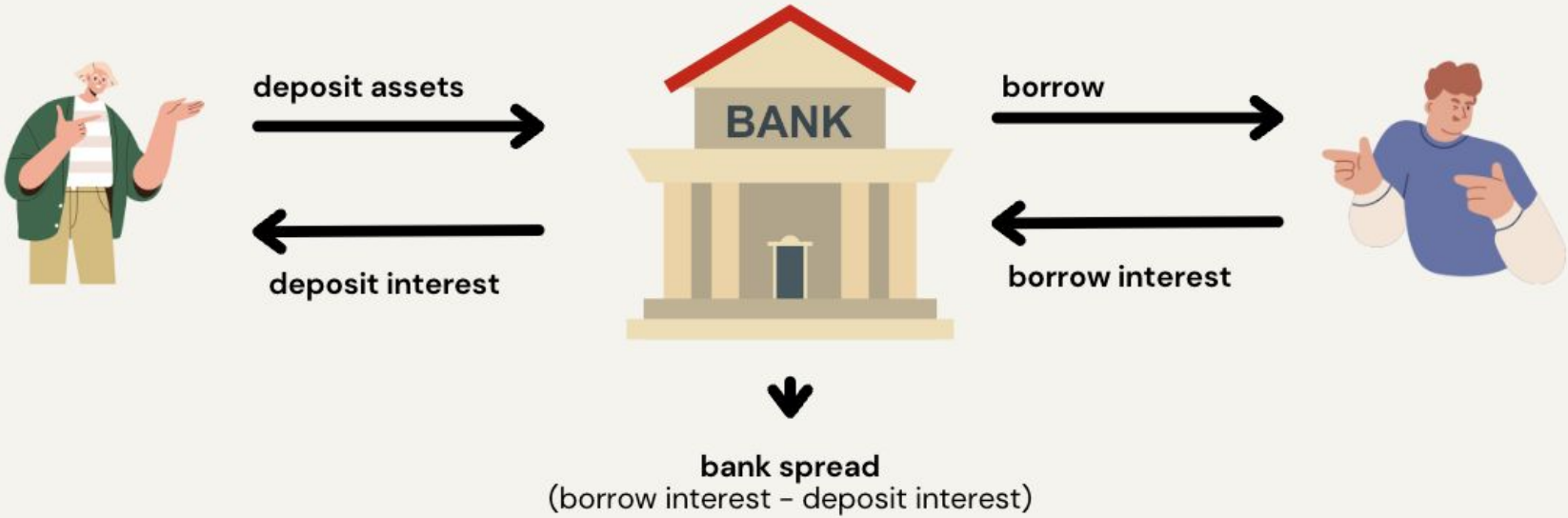
Acknowledgements

CS 251: Cryptocurrencies and Blockchain Technologies from Stanford University.

References to lecture slides:
https://cs251.stanford.edu/lectures/lecture9.pdf
https://cs251.stanford.edu/lectures/lecture10.pdf
https://cs251.stanford.edu/lectures/lecture11.pdf

Role of Banks in the Economy

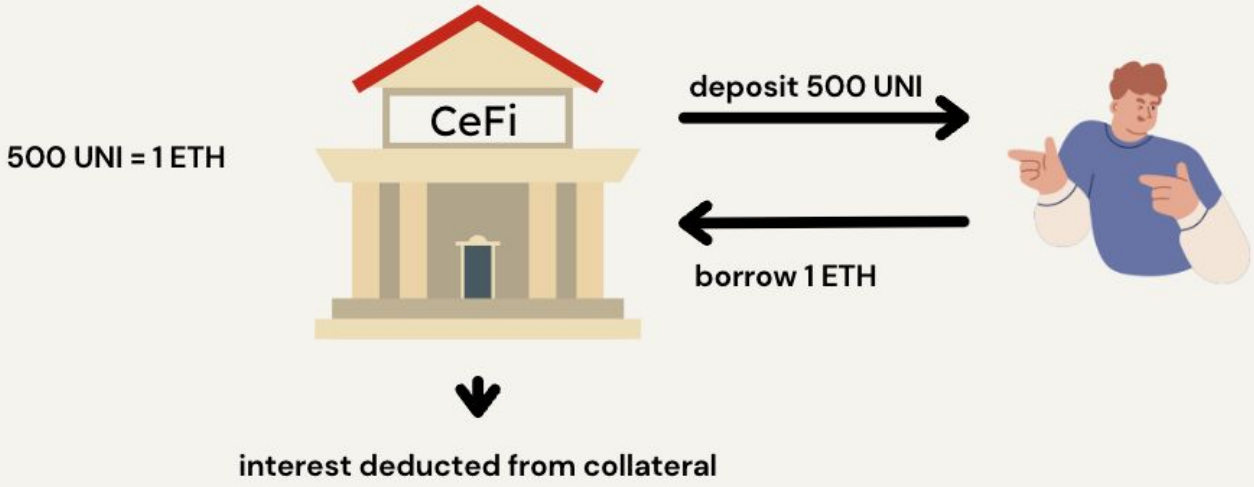
Banks bring together lenders and borrowers



Role of Collateral

CeFi's concern: what if Bob defaults on loan?
⟹ CeFi will absorb the loss.

Solution: Bob needs to lock up collateral.



Terminology

Collateral: assets that serve as a security deposit

Over-collateralization: borrower has to provide value(collateral) > value(loan)

Under-collateralization: value(collateral) < value(loan)

CollateralFactor $\in [0,1]$
- Max value that can be borrowed using this collateral
- High volatility asset $\Longrightarrow$ low collateral factor
- Relatively stable asset $\Longrightarrow$ higher collateral factor

Liquidation:
if value(debt) > 0.6 × value(collateral)
then the collateral is liquidated until inequality flips

Credit: Dan Boneh

Health of a Debt Position

$$\text{BorrowCapacity} = \sum_i \text{value}(\text{collateral}_i) \times \text{CollateralFactor}_i$$

(in ETH)

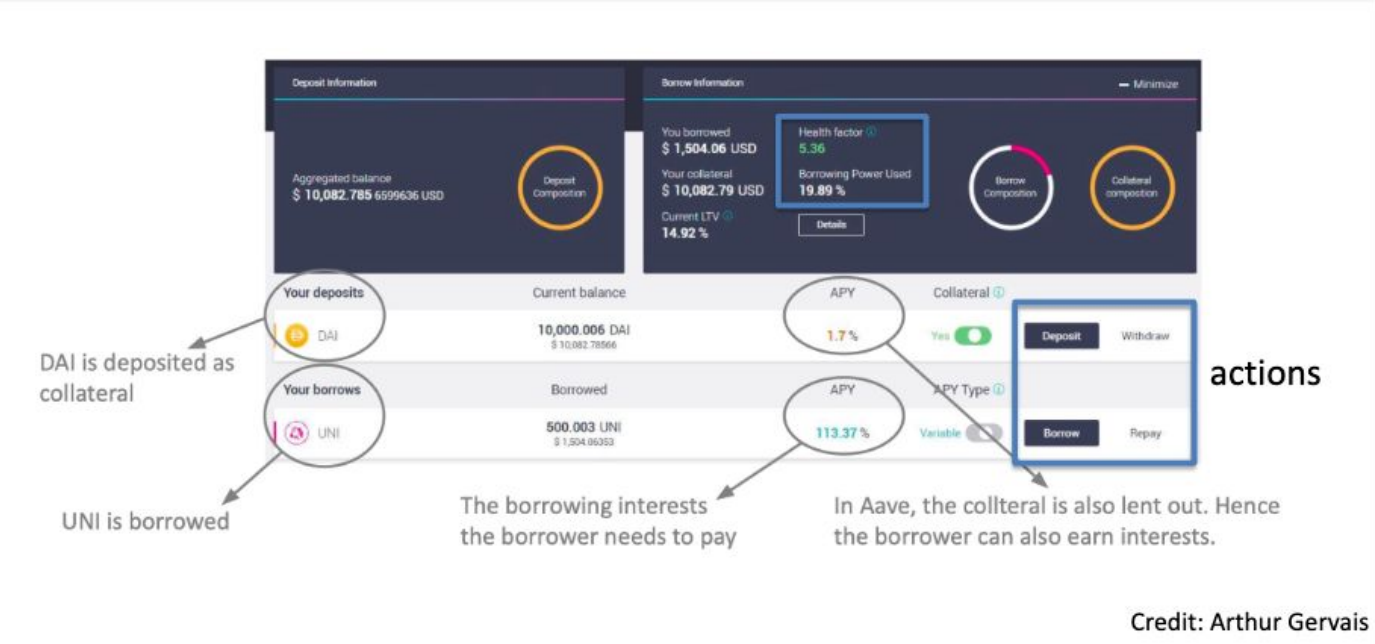
$$\text{health} = \frac{\text{BorrowCapacity}}{\text{value}(\text{TotalDebt})}$$

if: health < 1 $\Longrightarrow$ triggers liquidation until (health $\geq$ 1)

Credit: Dan Boneh

Aave Dashboard

(A DeFi Lending App)

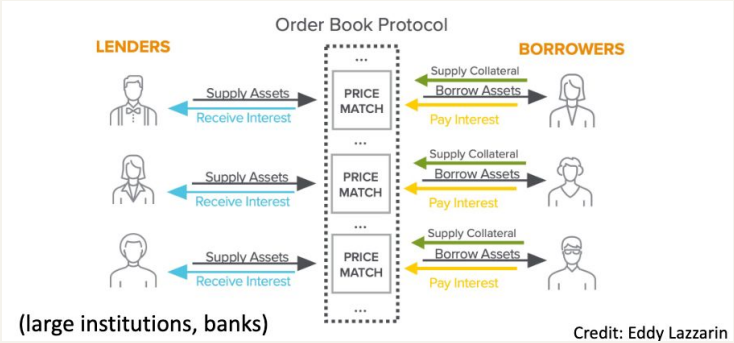


Credit: Dan Boneh

Decentralized Lending

Can we build an on-chain lending Dapp?
⟹ no central trusted parties
⟹ code available on Ethereum for inspection

Idea 1: An order book Dapp



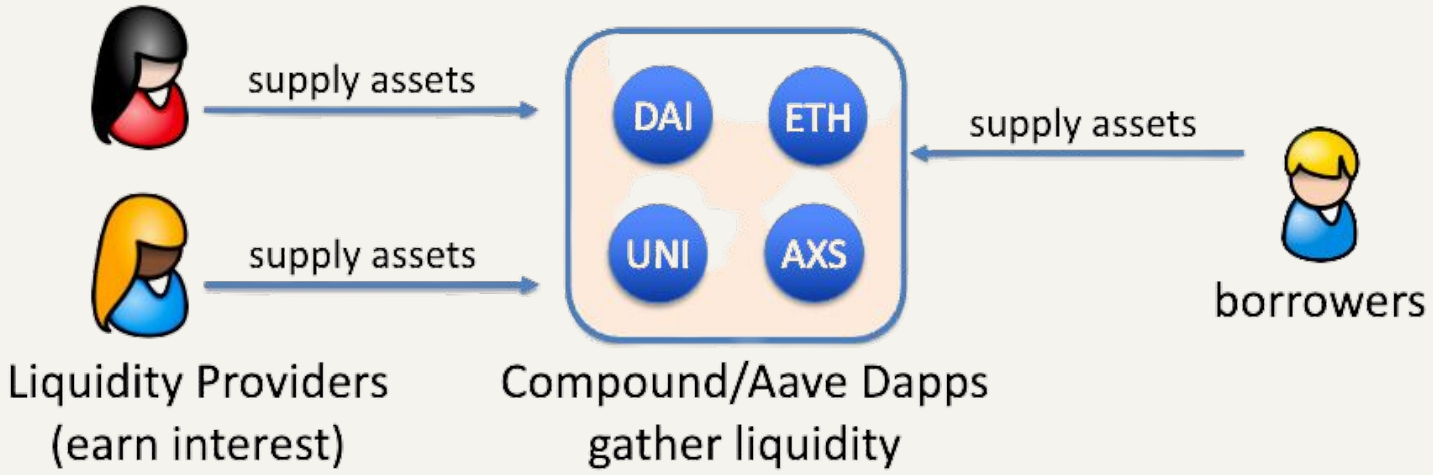
Challenges:
• Computationally expensive: matching borrowers tolenders requires many transactions per person
(post a bid, retract if the market changes, repeat)

• Concentrated risk: lenders are exposed to their direct counterparty defaulting

• Complex withdrawal: a lender must wait for their counter-parties to repay their debts

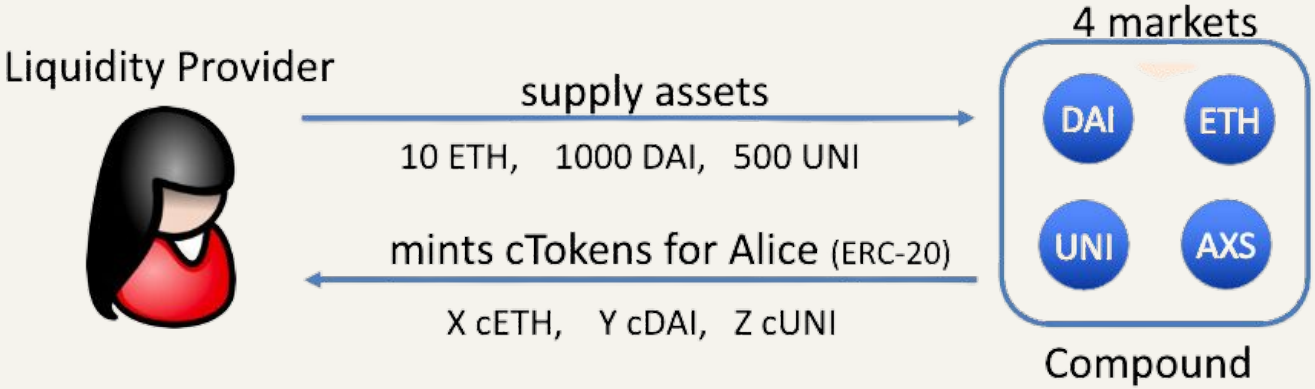
Credit: Dan Boneh

Liquidity Pools



Credit: Dan Boneh

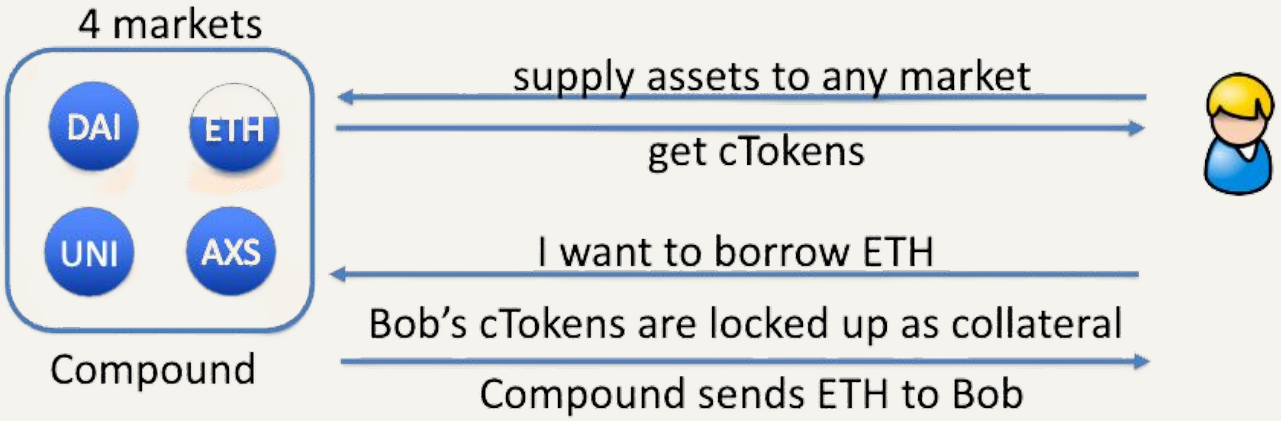
Compound: cTokens



Value of X, Y, Z is determined by the current exchange rate:
Token to cToken exchange rate is calculated every block

Credit: Dan Boneh

Borrowers



Bob's accrued interest increases ETH/cETH exchange rate

$\Longrightarrow$ benefit cETH token holders (ETH liquidity providers)

Credit: Dan Boneh

Exchange Rates

Consider the ETH market:

Supplying ETH: adds to $\text{UnderlyingBalance}_{ETH}$

Borrowing ETH: adds to $\text{totalBorrowBalance}_{ETH}$

Interest: added repeatedly to $\text{totalBorrowBalance}_{ETH}$

$$\text{ExchangeRate}_{ETH/cETH} = \frac{\text{UnderlyingBalance}_{ETH} + \text{totalBorrowBalance}_{ETH} - \text{reserve}_{ETH}}{\text{cTokenSupply}_{ETH}}$$

$\Longrightarrow$ As totalBorrowBalance increases so does ExchangeRate

Credit: Dan Boneh

Interest: Constantly Updates

Key idea: determined by demand for asset vs. asset market size

Utilization ratio: $$U_{ETH} = \frac{\text{totalBorrowBalance}_{ETH}}{\text{availableBalance}_{ETH} + \text{totalBorrowBalance}_{ETH}}$$

higher totalBorrowBalance, or
lower availableBalance in contract → higher $U_{ETH} \in [0,1]$

$$\text{interestRate}_{ETH} = \text{BaseRate}_{ETH} + U_{ETH} \times \text{slope}_{ETH}$$

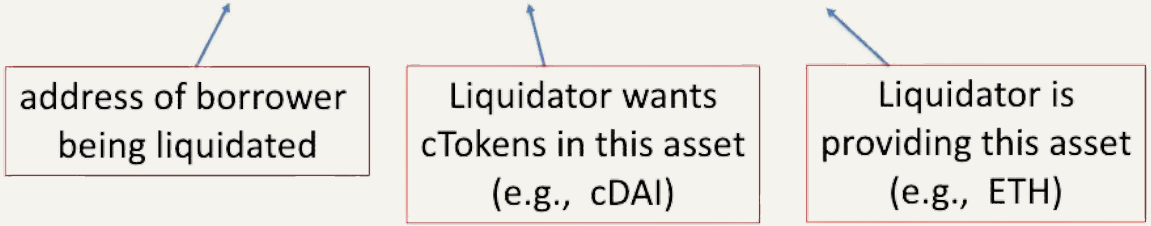
Credit: Dan Boneh

Liquidation

debt > BorrowCapacity

If user's health < 1 then **anyone** can call:

liquidate(borrower, CollateralAsset, BorrowAsset, uint amount)



This function transfers liquidator's ETH into ETH market,
and gives the liquidator cDAI from user's collateral

NOTE: Liquidator is repaying the user's ETH debt and getting the user's cDAI
[at a discounted exchange rate -- penalty for user]

Credit: Dan Boneh

Thanks!

Do you have any questions?