A Survey of DeFi Lending

COMS-6998-006 Fundamentals of Blockchains
Columbia University

Alex Brenebel (ab5181)
Lynsey Haynes (lah2224)
Vaibhav Kapur (vk2471)
Jonathan Larkin (jrl44)

December 18, 2021

1 Introduction

The act of borrowing and lending has been a fundamental primitive of finance for over 4,000 years. The earliest known record of lending dates from c. 2,000 BCE. At that time, in Assyria, India, and Sumeria, merchants, acting as prototype banks, gave loans to farmers and traders who transported goods along trade routes. In the Code of Hammurabi, c. 1750 BCE, various laws described the requirements of contractual terms for the repayment of a loan between a debtor and a creditor. The formal concept of a centralized financial intermediary, a bank, developed in Renaissance Italy in the wealthy cities of Florence, Venice, and Genoa. The oldest bank in the world still in operation, Monte dei Paschi, was founded during this period in 1472 [Wik21]. For over 500 years, the core function of commercial banking (making loans against deposits) has been the purview of special centralized institutions.

The publication of the Ethereum whitepaper in 2013 heralded the opportunity to enact lending transactions without peer-to-peer trust or a central trusted party. Ethereum, and other (near) Turing complete blockchains, have obviated the need for the centralized function of a bank, in certain transaction types, in favor of on-chain smart contracts. These decentralized finance, or "DeFi", lending protocols have matured and grown materially in the last year. In December 2020, there was $14bn in "total value locked" (TVL) in DeFi protocols; today there is approximately $105bn of TVL [def]. This paper is a survey of important and novel DeFi lending protocols, both over-collateralized ("OC" lending) and under-collateralized ("UC" lending). For convenience, when we say UC lending, we mean *both* "under-" and "uncollateralized" lending. We do not make the distinction between the two because, in the case of a blockchain, they both present the same key challenge: how to make a loan when the lender has risk which cannot be sufficiently covered by over-collateralization.

In the sections immediately following, we describe in detail the mechanisms of three collateralized lending protocols. We include discussion of attack vectors on the protocols themselves as well as attacks on price oracles used to measure and ensure collateral sufficiency. The protocols discussed are **MakerDAO**, **Aave**, and **Yield Protocol**.

Following the discussions of these three protocols, we move on to discuss the effort to bring UC lending to DeFi. Compared to OC lending, UC borrowers are motivated to seek UC loans to achieve better capital efficiency; UC lenders are seeking an opportunity to earn higher yields than available in the OC market. Since UC loans are riskier (there is risk of default without recourse to covering collateral), the market-clearing lending rates are naturally higher. In

the traditional finance world, UC lending (for individuals) rests on the incentive structure for the borrower to avoid the implicit costs of default, namely the reputation damage to credit scores, and the removal of access to financial infrastructure and mainstream employment. In this case, the borrower is a natural person who can be uniquely identified and tracked over time. Can this kind of lending exist on-chain? Blockchains are, in their most transparent form, pseudonymous. A natural person could, with minimal cost, hold access to many addresses and move funds between them or to new addresses at any time. This structure makes UC lending difficult to enact on blockchains. Nevertheless, there is significant development work being done in this area and some initial promising results. We discuss **Aave Flash Loans** and two other protocols, **TrueFi** and **Goldfinch**, which have executed material UC lending and repayment on-chain. We conclude this section with a short discussion of projects under development that could spur increased activity of UC DeFi activity.

2 Collateralized Lending

2.1 MakerDAO

MakerDAO is a Decentralized Autonomous Organization (DAO) built on the Ethereum blockchain using smart contracts [Makb]. MakerDAO brought the simultaneous innovations of allowing any user to borrow a stablecoin versus ETH collateral, facilitating this lending without relying on a centralized party, and creating an ERC-20 stablecoin, DAI, which maintains a peg to the US dollar based solely on the Maker algorithms and incentive structure. The protocol was launched on the Ethereum mainnet on December 18, 2017 and today its TVL is $17.9 billion USD [MKR]. This launch is considered to be the point where blockchains demonstrated material capability beyond payments. Today, Maker allows users to deposit other tokens in addition to ETH in order to mint DAI.

The Maker Protocol uses a two-token system. The first of which is Dai, a stablecoin soft-pegged to the US dollar and backed by Ethereum-based collateral assets. The second token is MKR, which is used by members to maintain the system through "Maker Governance". The MakerDAO is collectively run by MKR holders, and all activity, voting, and code is transparent and public. Voting power is based on the amount of MKR staked in a voting contract, so the members with the most voting power are the members who also have the most financial stake in the organization.

2.1.1 Loans in Dai

Users are loaned Dai by depositing Ethereum-based currencies, like ETH, as collateral assets in smart contracts called Maker Vaults. Dai are backed by excess collateral correlated to the risk parameter for that asset. A highly variable collateral asset has a higher risk parameter determined by the MKR holders, and therefore will require a higher deposit. Once a Vault is funded and accepted by the MKR holders, it is considered collateralized, and a collateral-to-debt liquidation ratio is set based on the collateral type. To withdraw the collateral from the Vault, the generated Dai must be repaid along with the stability fee, which is typically around 2-4% of the Dai generated. The loan does not have to be repaid on any schedule or timeline, and it is the user's responsibility to make sure that the value of the collateral does not fall below the liquidation ratio.

If the value of the collateral falls below the liquidation ratio, it is deemed too risky and can be liquidated by humans or more likely bots, called "Auction Keepers", to ensure there is enough collateral to cover the outstanding debt. Some of the Dai is paid to the Maker Buffer as a liquidation penalty and this penalty is used to encourage Vault owners to maintain adequate

collateral levels.

2.1.2 Collateral Pricing Mechanisms

The price of collateral is determined primarily by Price Oracles in a decentralized oracle infrastructure. MKR voters choose a broad set of individual nodes called Oracle Feeds to trust and deliver the collateral prices, as well as Emergency Oracles. The protocol first puts the price inputs through an Oracle Security Module (OSM), which delays the price information by one hour. In case an attacker gains control of a majority of Price Oracles, the Emergency Oracles or Maker Governance can vote to freeze an Oracle to protect the system. Emergency Oracles can also unilaterally trigger an Emergency Shutdown, in the event of an attack or a Maker Protocol system upgrade.

If collateral pricing from the Oracles becomes unavailable, the Maker protocol can do a collateral auction. The protocol takes the newly-liquidated Vault collateral and subsequently sells it in an internal market-based auction. If enough Dai is bid to cover the outstanding obligations plus the liquidation penalty, the auction converts into a "reverse collateral auction" to attempt to sell as little collateral as possible, and any leftovers are returned to the Vault owner. If there is a deficit, it is converted into Protocol debt and covered by Dai in the Maker Buffer. The Maker Buffer exists to pay off any potential Protocol Debt, and it is funded from liquidation penalties and stability fees. If the Maker Buffer gets too large and hits the threshold limit, its Dai gets auctioned for MKR tokens.

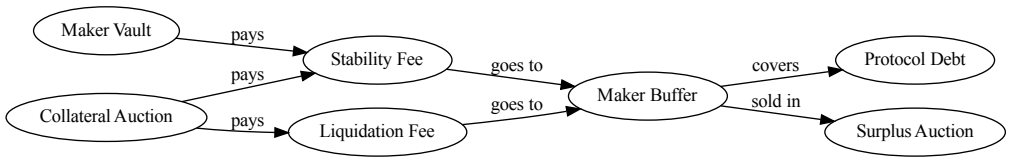

Figure 1: Maker Buffer Flow

There are three mechanisms to keep the market price of Dai pegged to 1 USD. First is the fact that Dai is backed by many cryptocurrencies in the form of collateral. Second is the global mechanism called the Dai Savings Rate (DSR). Dai holders can earn interest on their Dai by putting their Dai in a DSR contract. If the market price of 1 Dai dips lower than 1 USD, then the DSR rate can gradually be increased to create more demand and slowly increase the price of Dai. Vice versa, if the market price of 1 Dai is higher than 1 USD, the DSR can gradually be lowered in order to reduce demand, and slowly lower the price of Dai. The price of Dai has stayed relatively close to 1 USD since 2019, plus or minus 4 cents, so mechanisms appear to keep Dai at a stable price.

2.1.3 Growth and Scalability

While Maker first started as only accepting ETH, today it accepts over 25 cryptocurrencies as collateral. In September 2021, at a crossroads of decentralized and regulated finances, the bank Société Générale proposed using OTH tokens representing bonds backed by home loans to collateralize a $20 million dollar loan in Dai, to be repaid in 6-9 months [OTH21]. A collateral ratio of 130% with a liquidation ratio of 115% was proposed, but a major concern in the proposal was the potential illiquidity of the OTH tokens.

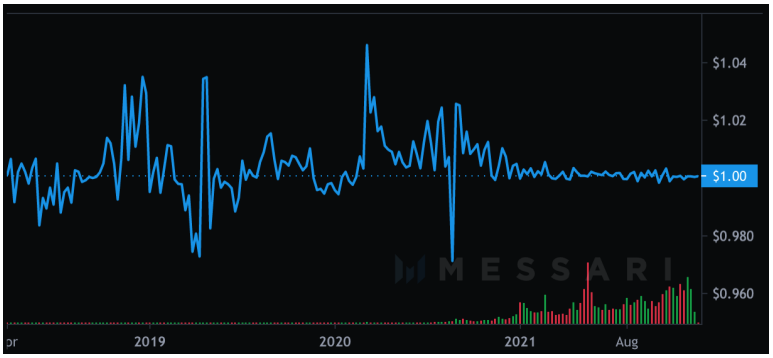

Figure 2: The Price of Dai Since Inception [Mes]

The computation required to calculate the amount owed on a Maker Vault or the amount held in DSR contract needs to be minimal in order to reduce Ethereum gas costs, and therefore needs to be be computed in constant time with respect to the current number of Maker Vaults or DSR contracts [Maka]. Maker achieves this by calculating a global cumulative rate value, which can be multiplied against a Vault's normalized debt to calculate the amount owed whenever the code needs it. For such a calculation, time is discretized into 1-second intervals, starting from t_0. The per-second stability fee is represented by F_i at time t, and the initial value of the cumulative rate is R_0. Then the cumulative rate is given by:

$$R(t) = R_0 \prod_{i=t_0+1}^{t} F_i$$

The normalized debt of a Vault is represented by A $= D_0/R_0$, where D_0 is the amount of debt at $t = 0$. Even though it is likely that the Vault did not exist at time zero, it is the amount of Dai that, if drawn at time zero, would result in the present total debt. Then to calculate the total debt D(t) of a Vault at time t:

$$D(t) = A * T(t) = D_0 \prod_{i=t_0+1}^{t} F_i$$

With this constant time operation, the Maker Protocol stores for each collateral type the cumulative rate, the total normalized debt across all Vaults, and the per-second rate. Each Vault stores only the normalized debt parameter, which allows the total debt owed to be calculated whenever needed. An interesting implication is there is no stored history of repayments or additions to the Vault, and no history of stability fee changes.

2.1.4 Risks and Mitigation

Dai is backed by many cryptocurrencies, and if a significant amount of one of those currencies loses a significant amount of value, it could trigger a cascade of liquidation events where the auction price of the collateral is not enough to cover the protocol debt. USDT, or the stablecoin Tether, is a currency that backs Dai, and while it was accepted as a collateral for Dai, Tether is not completely transparent about its backing, and some skeptics estimate only 2.9% of its supply is backed by cash reserves. To mitigate the risks around adding more forms of currencies, MakerDao has a proposal for new collateral assets that must be filled out before consideration by MKR holders. MakerDao also has a debt ceiling for each collateral asset, with the goal being that a total liquidation of that asset would not negatively affect the market value of the asset.

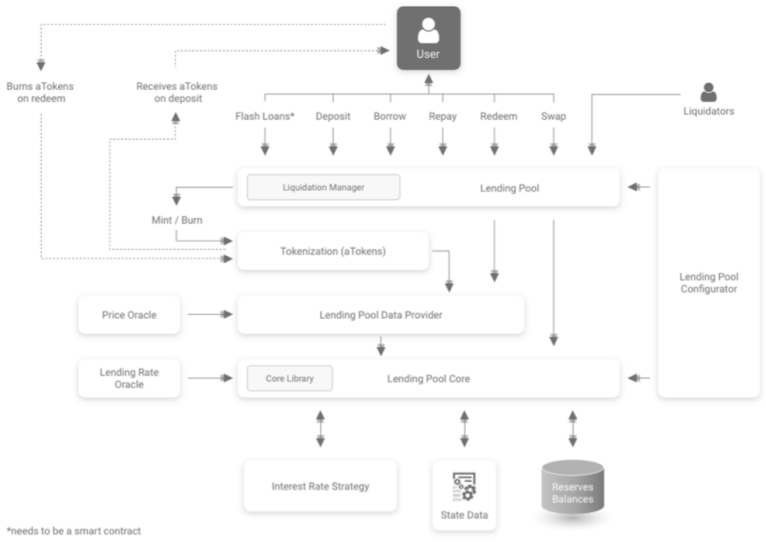

Figure 3: Aave v1 architecture [Aava]

2.2 Aave

Aave (https://aave.com/) is a decentralized lending system, running on the Ethereum blockchain, that allows users to lend, borrow and earn interest on crypto assets. The project was launched in November 2017 as ETHLend and later renamed Aave in September 2018. ETHLend was different from Aave in that, instead of pooling funds, it tried to match lenders and borrowers in a peer-to-peer fashion. It raised $16.2 million in an initial coin offering (ICO) in 2017, during which time it sold 1 billion units of its AAVE cryptocurrency - originally named LEND [Kra]. It was founded by Stani Kulechov and has a total value locked (TVL) of $13.76 billion [DLL]. The first version Aave v1 [Aava] was relased in January 2020, with the second version Aave v2 [Aavb] being released in December 2020. Aave enables users to lend or borrow 17 different cryptocurrencies including ETH, BAT and MANA [Kra].

2.2.1 Working

Evolving from ETHLend's peer-to-peer strategy, the Aave protocol has become a pool based strategy. Lenders deposit cryptocurrencies in a pool contract which enables borrowers to be able to take more loans from the pooled funds. The interest on these loans not only depends on the amount borrowed and collateral submitted by the borrower, but also on the state of funds in the pool. Hence, as more funds are borrowed from the pool, the amount of funds available decreases and the interest rate is raised. This pool-based strategy enables the ability to give instant loans to Borrowers. For lenders, the interest they earn on deposited funds corresponds to the earn rate, with the algorithm safeguarding a liquidity reserve to guarantee withdrawals at any time.

In this pool-based lending strategy, there is an important concept known as the **reserve**. Every lending pool has reserves in multiple currencies, with the total amount of reserves in Ethereum known as **Total Liquidity** [Aava]. The amount of loan that a Borrower can borrow depends on the currency deposit available in the reserve. Every reserve has a **Loan-To-Value (LTV)** [Aava] index, which is the weighted average of the different LTVs of the currencies composing the collateral submitted by the Borrower. The weight for each LTV is

the equivalent amount of the collateral in ETH.

Every borrow position can be opened with a stable or variable rate. In case the price of collateral drops below a threshold known as **Liquidation Threshold (LQ)** [Aava], a borrow position might be liquidated since reaching this ratio incentivizes liquidators to buy the collateral at a discounted price. Similar to LTV, every reserve has a specific LQ. The average LQ is the weighted average of the liquidation thresholds between all the currencies composing the collateral.

An index known as **Health factor** H_F [Aava] is used to determine whether the loan borrowed is undercollateralized:

$$H_F = \frac{TotalCollateral(ETH) * LQ}{TotalBorrows(ETH) + TotalFees(ETH)}$$

If $H_F < 1$, a loan is considered undercollateralized and borrower's position can be liquidated.

2.2.2 Architecture

The Aave v1 protocol architecture is shown in Figure 3. A key component of the architecture is the **LendingPool** [Aava] contract. The LendingPool contract uses the LendingPoolCore and LendingPoolDataProvider to interact with the reserves through actions such as Deposit, Redeem, Borrow, Repay, Rate Swap, Liquidation and Flash loans. The **LendingPoolCore** [Aava] contract is the center of the protocol. It holds the state of every reserve, holds all the assets deposited, and handles basic logic calculations (like calculating the value of interest rates). The **LendingPoolDataProvider** [Aava] contract performs calculations on a higher layer of abstraction than the LendingPoolCore and provides data for the LendingPool.

One of the features in lending pool is the **Tokenization** of lending position. When a user deposits in a specific reserve, the user receives corresponding amounts of **aTokens** [Aava]. These aTokens map the liquidity deposited and generate interest to the lender. They are minted upon deposit and their value increases until they are redeemed or liquidated. When a user applies to borrow, the tokens used as collateral are locked and can not be transferred.

Aave v1 brought a new innovation to the DeFi lending system known as Flash loans which will be discussed in more detail in section 3.1.

2.2.3 Transition to v2

There were challenges in the mechanism of Aave v1 such as the inability to upgrade the aTokens and gas inefficiency and this propelled a new version of the protocol: Aave v2. The architecture of Aave v2 is displayed in Figure 4. A major change in comparison to Aave v1 is that funds previously stored in the LendingPoolCore contract are now stored within each specific aToken. Another modification was the removal of LendingPoolCore and LendingPoolDataProvider and replacing them with libraries. This reduced the gas footprint of all the actions by 15 to 20%.

A key innovation introduced in Aave v2 is **Debt Tokenization** [Aavb] which entails that borrowers debt is now represented by tokens instead of internal accounting within the contract. This enables code simplification on the borrowing aspect of the protocol and allows simultaneous borrows with variable and multiple stable rates.

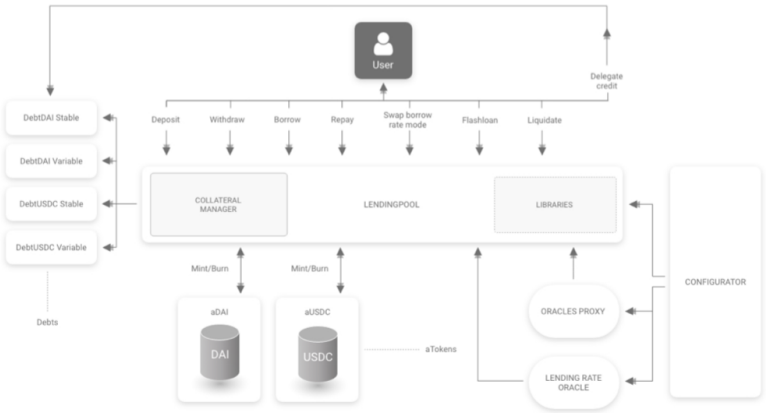

Figure 4: Aave v2 architecture [Aavb]

2.3 Yield Protocol

One of the most recent developments in the DeFi space is the Yield Protocol. Going live in late 2020, the Yield Protocol is a collateralized, Ethereum-based protocol specializing in fixed interest rates, fixed terms borrowing and lending. The main issue this protocol wishes to address is the volatility and lack of predictability in the DeFi floating-rate lending and borrowing markets.

In current DeFi DAOs and DAPPs, lending protocols are at the mercy of the crypto market's volatility. Maker DAO, Compound, Aave and many others have borrowing/lending systems in place to swap coins or tokens, with automatically calculated interest rates in place to ensure profit to the lenders. These interest rates are calculated based on the prices of the two assets (coins, tokens, etc.), and since there are high levels of volatility in the crypto marketplace, there will also be high volatility in the interest rates as well. This can lead to scenarios where the borrowers will lose money, even if their investments initially would have netted a gain. For example, let's say a user borrows DAI to buy some ETH with a 3% interest rate. That user makes a 7% increase and pays back the 3% interest rate, leaving them with a 4% profit gain. Now let's say they make an agreement with a 3% interest rate, only to have it change to a 10% interest rate. They still make a 7% increase but now have to pay 10% interest back, leaving them with a 3% loss. It was because of this unpredictability that the Yield Protocol was conceived; to create a system where interest rates are fixed.

2.3.1 Likeness to Zero-Coupon Bonds

The way the Yield Protocol aims to achieve this is through creating a system that acts similarly to Zero-Coupon Bonds. A Zero-Coupon Bond is a bond a person purchases at a cheaper price than what it is valued for, and after a fixed period of time, the bond reaches maturity and reaches its full value. The Yield Protocol acts as a standard for tokens that settle based on the target asset's value on a specified future date, backed by a collateral asset. The target asset is the asset in which the user wishes to borrow. The tokens that are being used are referred to as "fyTokens" or "fixed yield tokens".

The price of fyTokens float freely based on supply and demand, but will trade at a discount of their face value until their fixed expiration date is reached. Because we know the discount and the time to expiration, we can infer a yield (the annualized interest rate) that we would achieve when purchasing a fyToken and holding it to maturity. Let's say there are fyUSD

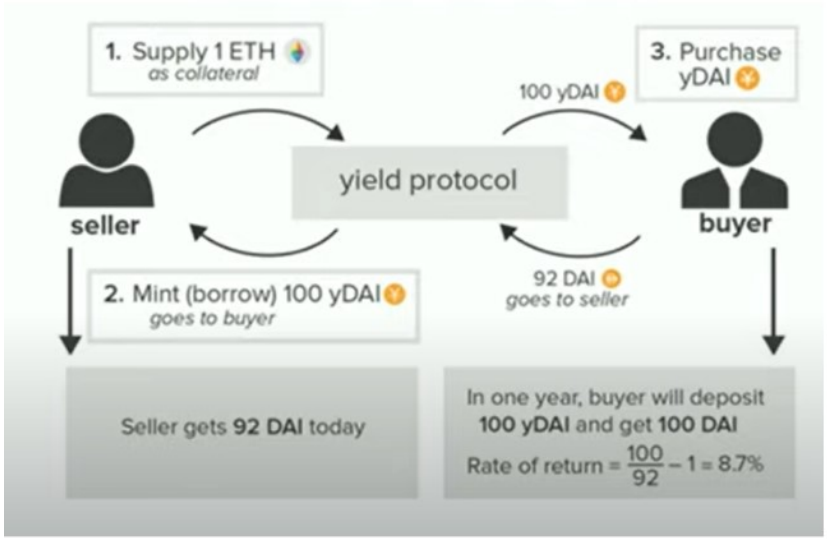

Figure 5: Yield Protocol Borrowing Mechanism [Har]

that have a face value of \$1 that are currently going for \$0.50. If a user invested \$1.00, bought two fyUSD and waited for maturity, they would have made \$2 and 100% gains. Therefore, the yield in this example would be 100%. The formula for calculating the annual yield is as follows:

$$S_n = Y = \left(\frac{F}{P}\right)^{\frac{1}{T}} - 1$$

where F is face value, P is present value, and T is number of years to maturity.

2.3.2 Borrowing and Lending

To create fyTokens, the user deposits a collateral and then sells the tokens to basically borrow the target asset. For example, let's say a user A wishes to borrow some DAI. They supply some amount of collateral, say 1 ETH, to the protocol. This creates 100 fyDAI tokens that they sell to a buyer B for 92 DAI. Now the buyer B has 100 fyDAI and the seller A has 92 DAI to use how they want. If we say this loan lasts for one year, we can calculate the fixed interest rate the borrower A has to pay to the buyer B who loaned the seller the DAI. $\frac{100-92}{92} = 8.7\%$ [Har]. By the end of the term (or fixed time when the contract finishes), we know that the borrower A owes 8.7% worth of interest, a fixed interest rate that does not change based on the volatility of the crypto markets. That is the basic premise for borrowing in the Yield Protocol.

With regards to lending using the Yield Protocol, buying fyTokens is economically similar to lending the target asset. Since fyTokens are not redeemable until expiration, they will most likely trade at a discounted price until maturity, especially if the target asset is in high demand [DR]. Let's use a simple example with DAI and fyDAI. A client X buys 100 fyDAI for 98.8 USD in September, with the the fyDAI maturing in December. Using the formula above, we can calculate that the client X will earn an implied rate of interest of 5% APR (T = 0.25 since 4 months is only a quarter of a year).

$$S_n = Y = \left(\frac{F}{P}\right)^{\frac{1}{T}} - 1 = \left(\frac{100}{98.8}\right)^{\frac{1}{0.25}} - 1 = 1.0494 - 1 \approx 0.05 = 5\%$$

2.3.3 Liquidity Providing

To improve the process of adding liquidity to the pools, the protocol was designed with a new Automated Liquidity Provider (ALP) that should enable efficient trading between Dai and fyDai called YieldSpace. In Uniswap, arbitrage trades are expected to occur when the price changes. In the YieldSpace Pool, arbitrage trades are expected to occur only when the interest rates change, which helps reduce the "impermanent loss" that market maker face (impermanent loss being "the temporary loss of funds occasionally experienced by liquidity providers because of volatility in a trading pair" [imp]). Users can provide liquidity to these pools to earn fees from future trades, using a fee model that is optimized for fyDai. Other ALP's charge a fee that is a percentage of the amount of an asset bought/sold, but YieldSpace charges an interest rate proportional to both interest rate and maturity time. The benefit to this fee model is that it makes sure the fees interest rates are fairly consistent and not overly wide-spread with regards to how much a borrower must pay back and how much a lender earns. This makes the fees more consistent and overall easier to manage since a user will generally know what to expect, reducing the effect volatility has.

One thing that is worth mentioning is how this protocol does not perfectly solve the problem of impermanent loss, rather solves time-dependent impermanent loss. When trying to liquidate fyTokens before their maturity date, the user runs the risk of suffering an impermanent loss. When you hold your fyTokens until maturity, you will not suffer any impermanent loss

2.3.4 Ties to MakerDAO

Another aspect to note is how the Yield Protocol relates to MakerDAO. The collateralization ratio is the ratio between the collateral value and the loan value. So for the examples above, that would be the value of collateral in Ethereum over the value of the loan in fyDAI. Both a Yield Vault's collateralization ratio and fyDAI's post-maturity borrowing/lending rates are determined by Maker's parameters. As such, those aspects can be altered by their governance system.

In addition, when checking the value of ETH in a Yield Vault, the Yield Protocol uses Maker's ETH oracle. Lastly, if there is an emergency shutdown triggered by the Maker protocol, borrowers/lenders in the Yield Protocol will also be affected.

$$\text{Collateralization Ratio} = \frac{\text{Collateral Value}}{\text{Loan Value}}$$

2.3.5 Settlement with Synthetic Assets

The Yield Protocol also has a system in place such that a user can also pay with a floating-point interest rate through making settlements with synthetic assets. A synthetic asset is a tokenized derivative that mimics the value of another asset, like DAI to the US Dollar. So when the target asset for a fyToken is a collateralized synthetic asset, the fyToken can use the synthetic asset's issuance mechanism for settlement. For example, user X can take owned fyDAI, backed by ETH, and when that fyDAI matures, the Yield Protocol can move the ETH collateral into a Maker vault. When user X comes to redeem the fyDAI, the protocol can borrow DAI from Maker and pay it out to user X. The protocol can also pay down any borrowers debt by releasing the borrower's collateral back to them. The borrower also has the option of placing their collateral and debt off into their own Maker vault.

Now, when a fyToken matures, the protocol needs to charge borrowers a floating rate to keep the debt position open, accounting for the potential stability fee it needs to pay if it borrows DAI from Maker. The protocol will also pay additional, floating rate yields

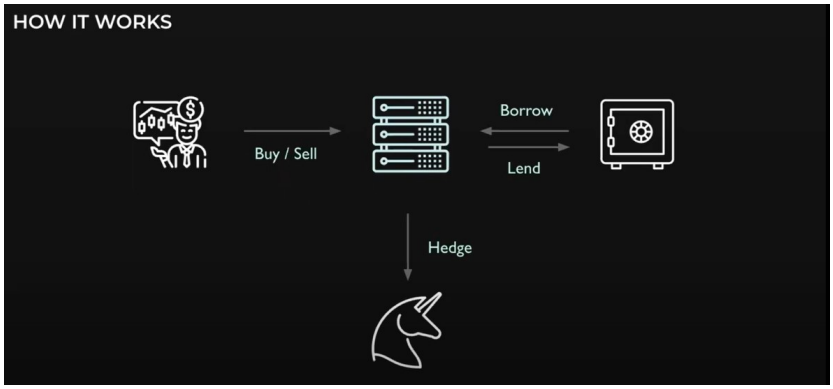

Figure 6: Contango's Expirable Future Diagram [Con]

to fyToken. With this understanding of the settlement mechanism, we can see this as users having fixed rate fyToken positions that are rolled into floating rate debt at maturity, since the protocol can charge borrowers the Maker stability fee while paying lenders their floating rate yields. The main advantage to this settlement mechanism is that both borrowers and lenders can continue to keep the same positions open after maturity. The only difference between other settlements is that here, the borrowers and lenders are now paying floating-rate interest rather than fixed, showing that even with the Yield Protocol, a borrower or lender can trade with floating interest rates after maturity.

2.3.6 Future Plans and Implementations

The engineers working on Yield Protocol have established several main goals in further developing the project. One of the main things they are working on is lowering the gas prices required to run Yield in their future iterations. One way they are aiming to achieve this is to create an L2 solution to try and improve the efficiency of computing transactions, though they are still going through a planning stage. Another aspect the team is aiming to improve are the quality of life improvements with the application itself, trying to convey the yields, returns, APY's (Annual Percentage Yields), and more different UI elements in a cleaner, easier to understand manner. The project is still in its early stages and the team has stated that they are focused on creating a useful working protocol, hoping that in several years they can hand over the protocol to the community [AN].

Recently a new development has come into play with regards to implementing the Yield Protocol through efforts to try and create a space for expirable futures in the DeFi market. Contango is a new project, created in Fall 2021, that wishes to create a DeFi counterpart to expirable futures that occur in the CeFi (Centralized Finance) market. With perpetual futures, there are no expiration dates and no control about the cost, meaning the future can last indefinitely and its value will vary depending on the market. This parallels the current systems with collateralized lending protocols that also don't have time limits and have yields be unpredictable and based on a volatile market (similar to Maker).

Contango plans to utilize the Yield Protocol's AMM (Automated Market Maker) to create a system, such that each time a trader buys or sells a contract, money is borrowed from a liquidity pool while a hedge is placed on a spot market (like UniSwap), with the hedge being lent to another pool (see Figure 5). This leads to borrowing and lending being done at a fixed rate by leveraging the YieldSpace smart contracts. The expirable futures basically take on the same properties of the zero-coupon bond, which motivated the Contango developers to implement the Yield Protocol's AMM.

These are the key concepts, features, and developments occurring within the Yield Protocol. While early in its development, the Yield Protocol has already been making progress towards lowering gas prices and gaining an audience. With the extreme volatility that comes with the crypto world, there is reason to believe that there will be a strong demand (especially for risk averse investors) for the Yield Protocol and other like-minded protocols emphasizing predictable and guaranteed interest rates and yields.

3 Uncollateralized Lending

As mentioned in the Introduction, UC lending presents significant challenges for DeFi. To motivate intuition, we begin our survey of UC DeFi lending by discussing the practical aspects of UC lending in the "real world." It is instructive to describe the process (at a high level) an individual undergoes when applying for the canonical UC loan: a credit card.

To start, the borrower makes a formal application. The applicant will note, at a minimum, the source and amount of his/her primary income, and personal identifiable information (PII) such as a social security number and physical address to allow the lender to uniquely identify the applicant. The lender will then access lending records held by a credit reporting company. Lastly, the lender will perform credit analysis and make a yes/no decision as to extend the credit to the borrower. After the credit is approved, the lender will make initial and monthly reports to the credit agencies as to the amount actively borrowed and the payment history (i.e., the degree to which the borrower has been timely with payments). If the borrower ceases payment, the lender will make an adverse report to the credit agencies; ultimately if the loan remains unpaid, it will be "charged-off" and recorded as such with the credit agencies. Under this common social arrangement, what is the incentive for the borrower to pay back the loan in a timely fashion? A credit report is bound to your identity for your life. Should you engage in activities, such as defaulting on a loan, which adversely affects your credit report, you suffer consequences. One financial press article cites eight adverse effects of a poor credit history including elimination of access to future lending options, increased lending rates in the future should you be able to secure a loan, lessened career opportunities (as employers typically check credit before making a hire), difficultly in renting a primary residence, and even difficulty in accessing basic utilities for your home [8Si].

A corporate borrower typically has access to two main sources of UC credit: the corporate bond market and borrowing from a bank. In the first case, the borrower issues an unsecured debt instrument which is purchased by investors. The investors make their credit decision by analyzing the financial strength of the issuer and the issuer's past credit repayment history. Credit ratings agencies take the place of the credit reporting agencies in the individual case. The borrower enters into a binding legal agreement (the "indenture") which specifies the terms of the bond. Importantly, in the case of default, the bond holder has the right to force the issuer into bankruptcy. In bankruptcy, the court will typically preside over an orderly liquidation of the company's assets; the unsecured lender is the last creditor in-line (behind the secured borrowers, but before the equity holders). Suffice to say, the executives and owners of a corporation (i.e., the equity holders) have significant incentive to avoid bankruptcy and therefore have a strong incentive to repay UC debts as expected.

The descriptions of UC lending in the real world should make one ask the question: *Can any loan be characterized as "uncollateralized" in the true sense of the word?* While these lending arrangements may not have physical or financial collateral, they have *social collateral*. Said differently, the borrower will undergo significant real difficulties in the case of default. When assessing the possibility of UC lending in DeFi therefore, a key question is: "is there some kind of intangible collateral, social or otherwise, which maps to the incentives borrowers in the real world have to repay their UC loans?"

3.1 Aave Flash Loans

3.1.1 About Flash loans

For loans in DeFi platforms, users have to deposit collateralized assets such as digital tokens. However, a new feature known as Flash loans enable non-collateral borrowing where an amount of asset can be lent to users by Flash loans as long as the borrowed assets can be paid back within the same, current transaction. Else, the transaction is reverted and lent assets are retrieved. For a Flash loan, users are required to develop a smart contract with the Flash loan provider [Fla].

The workflow of a Flash loan provider can be divided into five steps:
1. Flash loan provider transfers requested assets to User
2. Flash loan provider invokes User's pre-designed operation
3. User interacts with other contracts to execute operation with borrowed assets
4. Once execution is complete, User has to return the asset with the extra fee charged by the provider
5. Flash loan provider checks balance. If it is discovered that non-sufficient assets are returned by the User, transaction is reverted immediately.

3.1.2 Working in Aave

In Aave, Flash loans enable users to borrow from the reserves within a single transaction on the condition that users return more liquidity than taken. On initialization, Flash loans temporarily transfer the funds to a smart contract that respects the *IFlashLoanEnabledContract.sol* interface. After funds are transferred, *executeOperation()* is executed on the contract. The contract can then do any necessary actions with the borrowed funds. After *executeOperation()* is completed, a check is performed to verify that the funds plus fee have been returned to the lending pool. The fee collected is added to the reserve. In case funds less than what was borrowed was returned, the transaction is reverted [Aavb]. Once the Flash loan is invoked successfully, it emits a unique event called FlashLoan. This feature can be used to identify Flash loan transactions from Aave. As of January 2021, there existed over 15,000 Flash loan transactions in Aave [Fla]. Flash loans take advantage of a feature of all blockchains that transactions only get finalized when a new bundle of transactions, known as a block, is accepted by the network. Adding each new block takes time. On Bitcoin, that interval is approximately 10 minutes. On Ethereum, it's 13 seconds. An Aave flash loan therefore takes place in that 13-second period [Kra].

3.1.3 Applications

Flash loans were a key new innovation by Aave V1. It enabled users to perform actions such as refinance, collateral swap, arbitrage and liquidate. However, there is drawback that it can not be used with any of the other functionalities of the protocol. This was rectified in Aave V2 which enabled new opportunities such as collateral trading, repaying a loan with the collateral, margin trading, debt swap and margin deposits.

Collateral trading

The Aave protocol V2 offers a way of swapping assets deposited- both collateral and otherwise. Here, the *flashLoan()* function is called by the user which implements the *IFlashLoanReceiver* interface. This list is verified by the user in advance and contains an encoded list of underlying assets to swap from, a list of amounts of those assets, asset to swap to and the max slippage

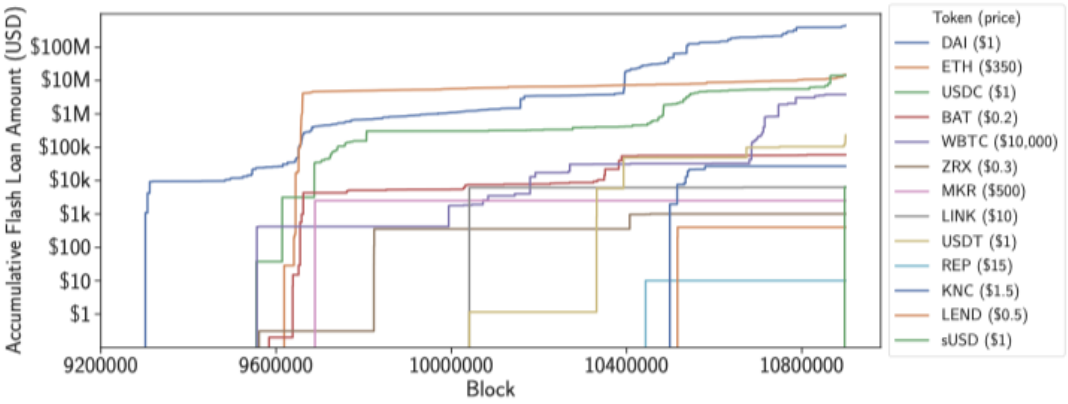

Figure 7: Accumulative flash loan amounts of 13 cryptocurrencies on Aave [Emp]

chosen by the user. The receiver contract uses these funds to swap them to the destination asset, i.e. depositing on behalf of the user and withdrawing user's deposits to repay the Flash loan [Aavb].

Repay with collateral

The Aave V2 protocol allows users to use assets deposited as collateral in the protocol to repay any debt. The repayment with collateral uses *flashloan()* and a receiver contract which implements *IFlashLoanReceiver interface*. The receiver is passed a list collateral assets as input. This is used by the receiver to swap and repay the amounts of those assets. This list also contains an encoded list of assets used to repay debt, a list of debt amounts to repay, and a list with the borrow modes for each debt asset. Here, receiver contract expects to receive an exact amount of how much needs to be repaid. This is in contrast to Swap Liquidity, where the amount expected is the exact collateral to swap [Aavb].

Arbitrage

In DeFi, Arbitrage is a behavior to gain benefits by trading between platforms supplying different price for an asset. As DeFi market reacts slower for events happening in the network than the real-world market, traders can take advantage of the market's inefficiencies to buy and sell the cryptoassets at a different price to gain financial benefits. Using Flash loans, traders can launch arbitrage without any pre-owned asset. If a price difference is found, the traders can instantly borrow a considerable asset with Flash Loan service to earn benefits. Thus, the Flash loan effectively becomes cost free with only the gas fee to be paid [Fla].

Wash Trading

In DeFi, wash trading is a group of trades increasing the trading volume on the asset or platforms. It can easily mislead users to perform financial operations on the targeted cryptoassets and platforms. While countries such as USA have banned wash trading from its markets, it is brought back to crypto market again because of the popularity of cryptocurrency and the lack of legal management. With Flash Loans, the wash traders can manipulate the market without a large amount of capital as long as they can afford the potential loss and the gas fee [Fla].

Table 1: Striking events exploiting Flash Loan

Date	Event Label	Flash Loan Provider	Implicated DeFi Protocols	Type	Proceeds
2020-02-15	bZx Pump Attack	*dYdX*	*bZx* *Compound* *Kyber* *Uniswap V1*	Pump and Arbitrage	$330K
2020-02-18	bZx Oracle Attack	*bZx*	*bZx* *Uniswap V1* *Kyber* *Synthetix*	Oracle Manipulation	$638K
2020-06-28	Balancer Attack	*dYdX*	*Balancer* *Uniswap V2*	Pump and Arbitrage	$439K
2020-10-26	Harvest Attack	*Uniswap V2*	*Harvest* *Curve* *Uniswap V2*	Oracle Manipulation	$26.6M
2020-11-06	Cheese Bank Attack	*dYdX*	*Cheese Bank* *Uniswap V2*	Oracle Manipulation	$3.3M
2020-11-12	Akropolis Attack	*dYdX*	*Akropolis*	Reentrancy	$2M
2020-11-14	Value.DeFi Attack	*Aave* *Uniswap V2*	*Value.DeFi* *Curve* *Uniswap V2* *SushiSwap*	Oracle Manipulation	$7.4M
2020-11-17	OUSD Attack	*dYdX*	*OUSD* *Uniswap V2* *SushiSwap*	Reentrancy	$7.9M
2020-12-18	Warp Finance Attack	*Uniswap V2* *dYdX*	*Warp Finance* *Uniswap V2* *Sushiswap*	Oracle Manipulation	$941K

Figure 8: Striking events exploiting Flash Loan [Attb]

Flash Liquidation

Liquidation is a behavior launched by the liquidator to buy undercollateralized assets from the lending platforms. Some lending platforms allow liquidators to compete on the keeper's undercollateralized assets like an auction. The winners, who pay the higher gas fee to launch their transactions, can buy the undercollateralized collateral with a discount. With Flash Loan, anyone can become a liquidator to make profits without much capital by buying the undercollateralized assets with a specific discount [Fla].

Collateral Swap

In DeFi, Collateral swap is well-defined behavior consisting of two main steps- Swapping and Operating. Swapping means redeeming collateral from the old loan and Operating means launching operations on redeemed collateral. Because the crypto market is extremely unpredictable, timely closing an existing collateral position is essential for a holder so they can stop losses from occurring through severe slippages and liquidations. For users without sufficient capitals for Swapping, Flash Loan can solve their urgent need by providing cost-free assets to save their collateral from the price slippage and the liquidation. They also enable Swapping and Operating actions run within one transaction. It further prevents users from suffering the uncertainty like slippage between transactions [Fla].

In Figure 7, the accumulative flash loan amounts of 13 different loan currencies on Aave are shown(Jan 2020 to Sep 2020) [Emp]. Among them, DAI is the most popular with the accumulative amount of 447.2M USD. The Aave flash loan transactions were inspected and classified depending on which platforms the flash loans interact with. It was noticed that the flash loan's transaction costs (i.e. gas) appear significant- at times beyond 4M gas compared to 21k gas for regular Ether transfer. The dominating use case was arbitrage and liquidation.

While Flash loans provide convenience, it also enables attackers to launch malicious operations with a large amount of asset that they do not have. Flash loans enable anyone to have instantaneous access to massive capital. It can either bring traders benefits or be used maliciously. As shown in Figure 8, several influential events exploiting Flash Loans have occurred since February 2020. The two attacks that happened in February 2020 were detailed and categorized by Qin et al [Atta] as Pump and Arbitrage attack and Oracle Manipulation attack respectively. Following this notion, it can be observed that most of the attacks listed in the figure correspond to a manipulation of oracles. The attack listed on November 14, 2020 where Aave was the Flash loan provider too is of type Oracle Manipulation [Attb].

3.2 TrueFi

TrueFi (`https://truefi.io/`) is the 11th largest DeFi lending protocol with $987mm in TVL [DLL]. The protocol was launched on the Ethereum mainnet on November 21, 2020. TrueFi has a public roadmap [TFR] and is in the middle of phase three of four of its roll-out. Per its app main page, TrueFi boasts that it has issued over $1bn in uncollateralized loans on-chain with a 0% default rate. This feat alone makes it worthwhile to study. Notably, the average loan size is approximately $5mm. The protocol is built with the idea of "progressive decentralization" [TFS]: TrueFi hopes to bootstrap a fully decentralized end-to-end UC lending flow, but today, it is a mix of DeFi and traditional lending.

While anyone can, in a permissionless fashion, act as a lender in TrueFi, loans are only made to vetted and white-listed institutional borrowers. This choice was made because large institutional borrowers with public profiles would incur substantial intangible cost in the case of default. To date, these borrowers are trading and investment firms. The on boarding of a borrower is similar to the first stage of the real world credit approval process. TrueFi verifies the borrower's identity, performs know-your-customer (KYC) checks, performs anti-money-laundering (AML) checks, and makes a credit assessment through its Credit Committee. If the borrower is approved, its agent executes a legally binding Master Loan Agreement (MLA) with TrueFi as its counterparty. If the borrower defaults, TrueFi, theoretically, can go to the courts in the jurisdiction of the borrower to enforce its claims under the MLA. The MLA mandates dispute resolution in the State of California and a key input to the white-listing approval is TrueFi's assessment that the MLA is binding the borrower's place of incorporation. The agent of the borrower then introduces itself on a TrueFi forum (see Appendix A for an example forum post): the borrowing institution makes a post describing their firm, their activities, social media links, website URL, the kinds of trading strategies they pursue, their assets under management, and the proposed use of funds [**?**].

Clearly, up to this point in the process, there is no value-add from TrueFi being associated with a protocol on a blockchain: this is a permissioned process that relies on traditional identity and legal infrastructure to incentive and enforce repayment. While this is disappointing, it is fair to recognize that this protocol was only launched thirteen months ago and, once on-boarded, the borrower/lender interactions (i.e., the approval and granting of a specific loan, and the repayment process) *are* all on-chain.

The lending process follows the TrueFi workflow as depicted in Figure 9. A lender deposits one of three USD-pegged stablecoins into a lending pool: USDCoin (USDC), Tether (USDT), or TrueUSD (TUSD). Each of these coins is a centralized stablecoin purported to be backed by fiat cash or short term fixed income investments held by a custodian. Deposits which are not actively lent out to a TrueFi white-listed borrower are placed in one of a few possible OC DeFi lending pools (Curve is one example). In return, the lender is issued an ERC-20 token representing the deposit (e.g., "tfUSDC" when depositing USDC). These tokens are

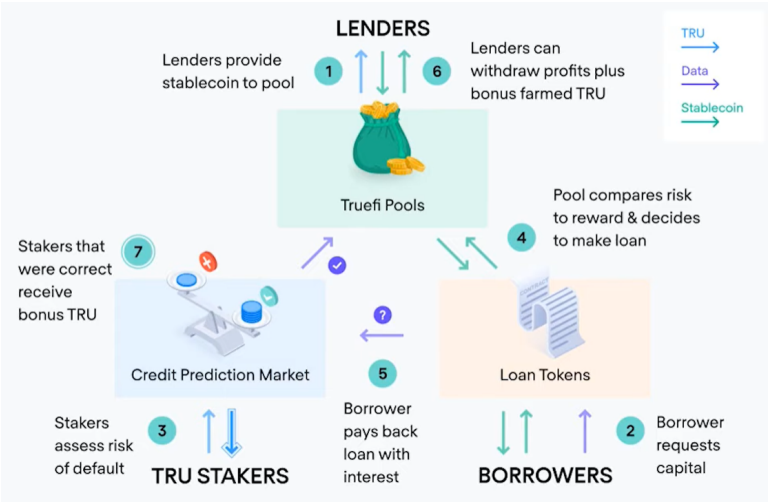

Figure 9: The TrueFi Lending Workflow [Trub]

not transferable, but earn a variable "farming yield" of TRU tokens and 90% of the interest paid on the loan. The TRU token is the native governance token of the protocol. It is a tradable ERC-20 token which allows holders to vote on governance proposals and stakers to vote on loan approvals and earn fees. This voting processing is optimistically called the "Credit Prediction Market", but is really just a simple voting process.

A white-listed borrower makes a request for a loan on-chain by minting a LoanToken and supplying the parameters of the loan: the term, the rate the borrower is willing to pay, and the amount. At this time, TrueFi only allows loans of 30, 60, and 90 days term in an amount from \$10k to \$20mm. Participants who have staked the governance token, TRU, then vote to approve or deny the loan. If approved, the funds are disbursed to the borrower on-chain and repayment (plus interest) is expected on-chain at the end of the term. TrueFi provides two interesting pieces of information to the stakers to help them make the lending decision: (i) a *TrueFi credit score*, and (ii) an estimate of the "fair" loan rate given the parameters of the loan and the credit score.

At present, the TrueFi credit score, ranging from 0 to 255, is a very traditional measure of creditworthiness and is assessed in a discretionary manner by TrueFi. Factors taken into account are company background, TrueFi repayment history, company operating and trading history, firm assets under management, and traditional credit metrics such as asset coverage, leverage, liquidity, and risk exposures [TFC]. This credit score is published on chain, bound to the borrower's address, and available via an API for, perhaps, other lending protocols to access.

The "fair" loan rate is produced by the TrueFi Rate Model. This model is not available to the public, but is transparent to TRU holders. This model was voted on and approved by TRU holders in a governance vote. If a borrower applies for a loan and specifies a rate which is materially below the rate produced by the Rate Model, it is expected that the TRU stakers will deny the loan.

Before exploring the important case of loan default, it is important to understand the incentive structures in TrueFi for the lenders and stakers. The TRU staker receives "staked TRU" ("stkTRU"), an ERC-20 token which allows the staker to participate in TrueFi's governance and to vote to approve or reject individual loan applications. Stakers receive 10% of all interest paid on loans, plus additional TRU tokens as an extra incentive, but are at risk for stake slashing of up to 10% per default. The TRU stakers bear the "first loss" of the loan: i.e., TRU from approvers are slashed up to the lesser of the full amount of the loss or the

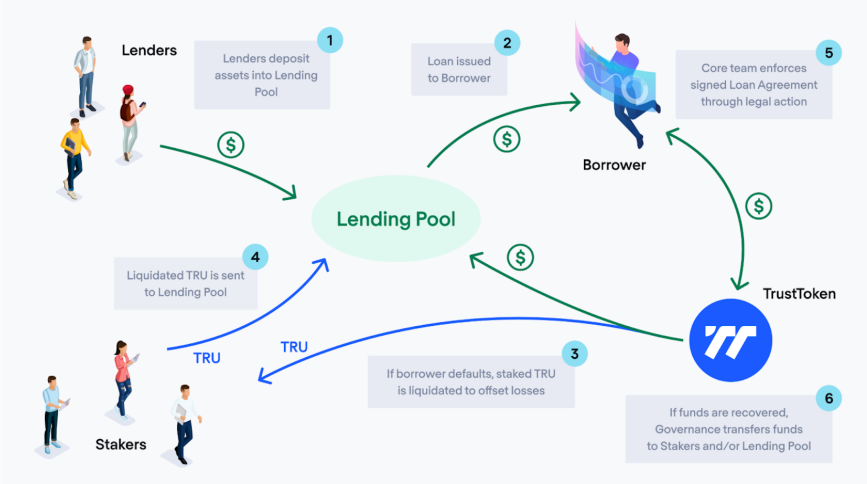

Figure 10: The TrueFi Default Workflow [Mit]

sum of 10% of each approver's stake. The lending pool then bears the remaining loss. This incentive structure is intended to be balanced: stakers have incentive to approve loans to earn staking rewards and interest, but have the incentive to be prudent to avoid stake slashing.

A natural question for a UC lending protocol is "what happens in the case of a default?" Given that TrueFi today is a mix of traditional lending and DeFi, the default process is complex. The default flows are visualized in Figure 10.

At the time of default, up to 10% of the loss is seized from the TRU stakers (using a Chainlink price oracle to value the TRU). This value is transferred into the lending pool. The loan is then marked down to 0 and this write-down affects the price of the pool token (e.g., the tfUSDC token if the loan is denominated in USDC) proportionally to the value of the defaulted loan to the pool. The TrueFi core team then pursues legal action to recover the debt. In the case that legal action recovers value from the borrower, the TRU stakers get the recovered value [Mit]. One observation is that the recovery process is quite unusual in that the senior lender has a binary outcome: the lender either gets full repayment with interest, or zero. This is not consistent with "real world" bank lending where the senior lender shares in the recovery value.

The TrueFi process today is not scalable; however, the long-term vision for TrueFi is that the staking process becomes sophisticated enough to take over as much of the manual screening and on-boarding process as possible. It is notable that every loan the protocol has made is transparent and on chain and that borrowers are building an on-chain credit score. TrueFi endeavors to migrate its process to be as decentralized and on-chain as possible. TrueFi founder and CEO, Raphael Cosman, envisions an evolution of TrueFi where TRU stakers do not vote on every loan, but instead vote on the rate model and credit scoring model. In that setup, loans would automatically be approved if they pass the screen of the automated models. TrueFi also envisions that its credit score can eventually solve the "credit rating for DeFi" problem [Trua]. The ethos and promise of DeFi is a permissionless and open replacement of traditional financial intermediaries. TrueFi is far from this, but it's accomplishments to-date are impressive and it is a protocol to watch closely.

3.3 Goldfinch

Another UC lending protocol which has good prospects for success is Goldfinch (`goldfinch.finance`). Goldfinch was founded by Michael Sall and Blake West and launched on the

Ethereum mainnet in December 2020 in a "private beta." The protocol currently has TVL of $37mm [DLL]. While this amount is minuscule compared to, e.g., the TVL of Aave, the Goldfinch approach has a good possibility to bring a legitimately decentralized approach to UC. First we discuss what Goldfinch is doing today, and then their road map for growth. In discussing Goldfinch's activities, it is important to keep in mind that this protocol is only a year old and is undergoing constant development and transformation. Primary sources for understanding Goldfinch become outdated quickly. We have attempted to triangulate to make an accurate portrayal of Goldfinch from various Medium posts, YouTube videos, the Goldfinch whitepaper, and threads and FAQs in the Goldfinch Discord channel. These references are noted in the text where appropriate. The Goldfinch team did not respond to our request to engage with them directly.

Today, Goldfinch focuses on UC lending to small- and medium-sized business in developing countries. This addressable market is very large: the International Finance Corporation (IFC) estimates that 40% of micro- to mid-sized businesses across all developing countries have unmet financing needs totalling over $5 trillion annually [Wor]. While the team has aspirations to build a global lending protocol to service both institutions and individuals, like TrueFi, they recognize that the building blocks for broad success in UC in DeFi are not there yet. They hope to bootstrap UC in DeFi with a careful growth plan. So far, the Goldfinch protocol has made loans to small businesses in Nigeria, India, Indonesia, Thailand, Vietnam, and Mexico. The team observed that there was a gap in "middle market" lending in these countries. Generally, local lenders exist which provide micro- and small loans up to $100k; multi-national institutions begin to lend at loan sizes of at least $5mm. Goldfinch is initially seeking to compete in the gap between $100k and $5mm [Cry]. The approach Goldfinch is taking today is philosophically similar to the TrueFi approach: keep the lending decision in the traditional finance world, and source the funds for loans on-chain.

Specifically, Goldfinch is acting as a wholesale lender and has partnerships with four traditional middle market lenders who are active in the named countries. These lenders do all the traditional loan underwriting steps and when a loan is approved, the documentation has the borrower facing a special-purpose-vehicle (SPV) in the relevant jurisdiction which is owned by Goldfinch. Goldfinch then sources USDC on-chain from lenders to fund the loans. It is unknown how risk-sharing works between the originator and Goldfinch. It is clear though that this end-to-end process gains almost nothing from being on-chain. However, the Goldfinch plan, as described in the Goldfinch whitepaper presents some novel ideas of how to bootstrap this approach into a true decentralized UC lending protocol. It is important to remember that UC lending is nascent in DeFi and, as such, we need to pay attention not just to what is precisely being done today, but ideas and plans from well-funded and credible teams. Goldfinch is one such team.

The Goldfinch whitepaper describes a protocol plan with five participants: Borrowers, Backers, Liquidity Providers, Auditors, and a Unique Entity Checker. These participants operate in a way that creates the proper incentives to facilitate UC DeFi lending with risk control and fraud prevention. In this aspirational plan, Goldfinch has *no* manual interaction in any decisions or flows–it simply provides the relevant smart contracts and issues the protocol's utility/governance token.

The envisioned workflow, shown in Figure 12, is as follows: a borrower publishes a proposal to take out a loan; this proposal specifies the amount, interest rate, payment schedule, and term and is called a "pool" in the protocol. This pool is an instance of a smart contract and is seen by the community of Backers. The Borrower must stake a number of Goldfinch governance tokens, GFI. The amount is fixed by the protocol; half of this stake is transferred to the the Auditors (described below) as staking rewards and the other half is returned when the loan is paid in full. If the loan is not approved, the stake is claimed by the protocol.

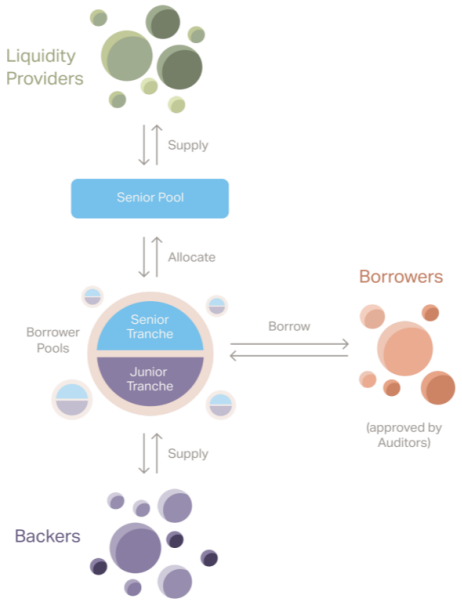

Figure 11: Planned Goldfinch Lending Workflow [Gola]

The amount of GFI staked at this point by the Borrower is presumed to be substantially smaller than the principal value requested in the loan in order to retain the "uncollateralized" intention of the protocol. This GFI stake exists to avoid Borrowers "spamming" the protocol with loan requests. An honest Borrower will only make a loan request which is *bona fide* and which it believes has a good chance of being approved.

The Backers then decide, individually, whether to "back" the loan. To back a loan means to commit to providing "first loss" capital on the loan. This first loss capital is, in Goldfinch and traditional financial parlance, called the "junior tranche" of the loan. The Liquidity Providers fund the remaining principal of the loan; this remaining amount is called the "senior tranche" of the loan. In the case of default, the junior tranche bears the full loss until it is wiped out. If the junior tranche is wiped out, then the senior tranche bears the remaining loss. In exchange for backing loans and taking first loss risk, a backer earns an portion of the loan interest. The interest payments are split as follows: 20% to the junior tranche, 70% to the senior tranche, and 10% is retained by the protocol and managed through governance. The ratio of funding provided by the senior and junior tranches is a function of the number of Backers who participate. This is described formally below, but the general idea is that the number of Backers who participate is an information signal which is inversely proportionate to the true risk of the loan. Lastly, the Auditors are a community who own and stake the Goldfinch governance token, GFI. For every pool, a set of nine Auditors is randomly chosen; randomization is proportionate to stake (similar to, e.g., how Ethereum 2.0 block validators are chosen under Proof of Stake). The Auditors "vote" to allow the pool to move forward in the process. The Auditors are not making a credit decision; rather they are opining on if the Borrower is simply *bona fide*. An individual Auditor's stake is slashed if it votes *differently* than the consensus among the Auditors overall. An Auditor earns staking reward when it decides with the consensus.

Not shown in Figure 12 is the Unique Entity Checker ("UEC"). The UEC is an oracle (one or more) which certifies that the addresses belonging to the Borrower, Backers, and Auditors are all unique natural persons in the real world. This is imperative to mitigate Sybil attacks. Currently, the UEC is a company called Persona (`https://withpersona.com/`)

[Golb]. Persona provides KYC/AML as an API and cites significant FinTech companies such as BlockFi, Square, and Gusto as clients. Since protocol launch, Goldfinch has verified identities of 33k people [**?**].

The whitepaper contemplates the UEC(s) to eventually be on-chain contracts once the concept of a "decentralized identity" matures [Gola]. A month ago, Goldfinch announced their own solution for this – the Unique Identity NFT (UID). The UID is an ERC-1155 contract. To obtain an UID, an entity still interacts with Persona through a Goldfinch webpage, the user can mint a non-transferable NFT that is sent to the entity's Ethereum address. Due to the open nature of the Ethereum blockchain and the inherent composability of smart contracts, this UID could be used by any application to mitigate Sybil attacks. [Sal].

The Goldfinch processes are designed to be neither anonymous nor pseudonymous. It is envisioned that the Backers and the Auditors will perform credit diligence on the Borrowers. This process has the flavor of traditional lending, but it is truly decentralized. Backers, Liquidity Providers, and Auditors can enter and leave the community at any time and Goldfinch *does not* participate in any decision making [Cry]. As mentioned, a participant in the system does need to establish a unique identity through KYC/AML checks so the system is not permissionless.

The *Leverage Model* determines the leverage ratio which is the amount of capital provided by the senior tranche divided by the capital provided by the junior tranche. The Backers want leverage to be as high as possible as this allows them to commit the least amount of capital. Since the junior tranche earns a fixed 20% of the value of interest paid, the effective earned yield by the junior tranche is maximized when junior tranche capital is minimized. The USDC amount, A, that the senior pool contributes is calculated as $A = S \cdot D \cdot L$. S is the USDC value of the capital supplied by the Backers. The value of D is based on the "Herfindahl-Hirschman Index" which is an index of diversification and gives a value between 0 and 1.

$$D = 1 - \sum_{i=1}^{n} s_n^2$$

where s_i is the percent of total Backer capital contributed by Backer i. The protocol defines a value, (unspecified so far), for the minimum and maximum number of Backers allowable ($B_{\min}$ and $B_{\max}$ respectively) and a maximum leverage, $L_{\max}$. Given these protocol parameters, the permissible leverage of a pool is

$$L = L_{\max} \cdot \frac{\max(b - B_{\min}, 0)}{B_{\max} - B_{\min}}$$

The purpose of the Auditors is to guard against attack vectors on the protocol. One such attack is "Fraudulent Borrower and Backers"; in this attack there is collusion between the Backers and the Borrower. In fact, in a naive execution of the Goldfinch process, this attack could be a Sybil attack orchestrated by a single individual. The attack is as follows: the attacker, Bob, creates a large number of Backer addresses, cumulatively funded with USDC 10; he then creates a borrower address, and publishes a pool to request a loan for USDC 100. The backer addresses signal that they will back this loan and post the cumulative USDC 10. The loan is therefore executed and the remaining USDC 90 is provided by the Liquidity Providers. Bob then defaults on the loan and the USDC 10 in his Backer accounts (funding the the junior tranche) is wiped out. The Liquidity Providers lose the USDC 90 they provided in the senior tranche and Bob is better off from the attack by this amount. This attack is mitigated by the random selection of Auditors who have value at stake, and the Unique Entity Check, which makes it difficult to create the Backer Sybils.

Another attack is "Fraudulent Borrower and Auditors"; in this attack the Borrower colludes with the Auditors to get approval for a pool. This attack is mitigated by the random

selection of Auditors who have value at stake. In expectation, the Auditor Sybils must control $> 50\%$ of the consensus. In a situation where Goldfinch has scaled its protocol, this would be prohibitively expensive in comparison to the principal amount in any one pool. Finally, even if this collusion was successful, the Borrower would still need to convince the Backers to participate.

Lastly, the most basic attack is "Fraudulent Borrower and Honest Backers and Auditors". This is the key issue in UC lending: how to ensure the borrower is a good credit and pays back the loan. In Goldfinch, the Backers have financial interest to provide capital only to loans they believe are creditworthy, and the Auditors have stake value they want to protect and therefore have incentive to perform due diligence on the Borrower. While not required, it is intimated in various Goldfinch Discord threads that a Backer could enter into a traditional bilateral legal contract with a borrower, independent of the Goldfinch protocol. The creation of the UID and the open visibility of all transactions on the Ethereum blockchain could establish an address history as a "proof of creditworthiness". If an address is verified as a unique individual and it builds up a history of positive credit activities, this address becomes a store of social value.

The Goldfinch protocol is still in its early days. The lending activity on the protocol is relatively small and existing loans were likely sourced primarily through traditional channels. However, the Goldfinch protocol presents a novel incentive structure and process which promises to allow decentralized UC lending in DeFi as its ecosystem grows.

4 Conclusion

DeFi applications have been on the mind of blockchain visionaries, developers, and entrepreneurs since the early days of Bitcoin. In the Ethereum whitepaper, published in 2013, Buterin theorized about the benefits (well beyond supporting tokens as a store-of-value and for payments) that a Turing-complete blockchain could offer the world. He briefly described potential use cases including financial derivatives for hedging, stablecoins, decentralized autonomous organizations (DAOs), crop insurance, and price oracles (which he called simply decentralized data feeds) [Eth13]. Interestingly, he *did not* explicitly mention borrowing and lending! Although much of what he foresaw was indeed brought to implementation in some form, it is ironic that lending protocols have become the "killer app" of DeFi. Of all DeFi protocols, three of the top five protocols by multi-chain TVL are lending protocols: Aave, MakerDAO and Compound [DLL]. This survey explored the mechanisms of both Aave and MakerDAO (and we note that Compound is substantially similar in its implementation to Aave). We also explored Yield Protocol, a recent effort to bring fixed-rate OC lending to DeFi and establish something akin to the term structure of interest rates that we observe in traditional fixed income marketplaces. A related innovation to Yield Protocol is the YieldSpace automated-market-maker invariant. This invariant is an important innovation as it allows for more efficient pricing and trading of tokens in decentralized exchanges where an explicit time-value-of-money component exists.

Beyond simply replicating "real world" lending on-chain, DeFi has invented, in some ways, a new method to conduct finance. One example is Aave flash loans–a concept that does not map to anything preexisting in traditional finance. DeFi protocols are implemented in software and therefore exist as *composable* abstractions. This is a breakthrough in itself.

We are in the very early days of UC blockchain lending. In some ways, UC lending seems at odds with an often shared vision that ideal blockchains be fully open, permissionless, and censorship resistant. While the social value of blockchains may be maximized in that setting, these ideas do not all need to exist in order for blockchain projects to provide economic and efficiency value-add. The case in point for this is the current state of UC lending. The two protocols we described, TrueFi and Goldfinch, exist halfway between traditional finance and

true on-chain decentralization. They offer efficiency of capital access to a limited market of borrowers and dis-intermediate banks by allowing protocol participants to be the source of funds in a loan. The trend seems to favor the development of on-chain credit scores that are either linked permanently to addresses or are transparently calculated as a function of the on-chain activity tied to an address. In his whitepaper, Buterin did mention briefly "identity and reputation systems" as a potential important primitive and this idea is seen in the TrueFi credit scores and Goldfinch UID. DeFi OC lending is about two years old and UC is about one year old. The extraordinary progress in protocol development and growth in such a short amount of time is exciting. The pace of innovation does not appear to be slowing down and we expect significant further growth in DeFi lending in the coming years.

References

[8Si] 8 side effects of having a bad credit score. https://www.cnbc.com/select/side-effects-of-bad-credit/. (Accessed on 12/13/2021).

[Aava] Aave v1. https://github.com/aave/aave-protocol/blob/master/docs/Aave_Protocol_Whitepaper_v1_0.pdf.

[Aavb] Aave v2. https://github.com/aave/protocol-v2/blob/master/aave-v2-whitepaper.pdf.

[AN] Alberto Cuesta Cañada Allan Niemerg. Yield protocol discord server. https://discord.com/channels/752978124614008945/766766219750670377. (Accessed on 12/16/2021).

[Atta] Attacking the defi ecosystem with flash loans for fun and profit. https://arxiv.org/pdf/2003.03810.pdf.

[Attb] Flashot: A snapshot of flash loan attack on defi ecosystem. https://arxiv.org/pdf/2102.00626.pdf.

[Con] Contango exchange. https://contango.exchange/. (Accessed on 12/16/2021).

[Cry] A11 Crypto. Goldfinch finance - a decentralised credit platform for crypto loans without collateral. https://www.youtube.com/watch?v=35Wc5VP28fs&t=93s.

[def] Defi pulse - the decentralized finance leaderboard. https://www.defipulse.com/. (Accessed on 12/13/2021).

[DLL] Lending tvl rankings - defillama. https://defillama.com/protocols/lending. (Accessed on 12/13/2021).

[DR] Allan Niemerg Dan Robinson. The yield protocol: On-chain lending with interest rate discovery. https://yield.is/yield.pdf. (Accessed on 12/14/2021).

[Emp] An empirical study of defi liquidations: Incentives, risks, and instabilities. https://arxiv.org/pdf/2106.06389.pdf.

[Eth13] Ethereum whitepaper — ethereum.org. https://ethereum.org/en/whitepaper/, 2013. (Accessed on 12/17/2021).

[Fla] Towards a first step to understand flash loan and its applications in defi ecosystem. https://arxiv.org/pdf/2010.12252.pdf.

[Gola] Goldfinch whitepaper. https://goldfinch.finance/goldfinch_whitepaper.pdf. (Accessed on 12/13/2021).

[Golb] Goldfinch. Goldfinch discord faqs. https://www.notion.so/Discord-FAQs-79b9c077598f4c928c4b04aff0405a72. (Accessed on 12/16/2021).

[Har] Campbell Harvey. Yield protocol - mechanics. https://www.youtube.com/watch?v=uhJhARs_-f0. (Accessed on 12/14/2021).

[imp] What is impermanent loss? https://coinmarketcap.com/alexandria/glossary/impermanent-loss. (Accessed on 12/14/2021).

[Kra] What is aave? (aave). https://www.kraken.com/en-us/learn/what-is-aave-lend.

[Maka] Maker docs: Rates module. https://docs.makerdao.com/smart-contract-modules/rates-module.

[Makb] The maker protocol: Makerdao's multi-collateral dai (mcd) system. https://makerdao.com/en/whitepaper/. (Accessed on 12/12/2021).

[Mes] Messari dai price. https://messari.io/asset/dai. (Accessed on 12/14/2021).

[Mit] Mitigating risk & truefi's loan default process — by trusttoken — truefi. https://blog.trusttoken.com/mitigating-risk-truefis-loan-default-process-454359a8c4b. (Accessed on 12/13/2021).

[MKR] Makerdao mkr. https://defillama.com/protocol/makerdao/all/USD. (Accessed on 12/16/2021).

[OTH21] Mip6 application for oth tokens. https://forum.makerdao.com/t/security-tokens-refinancing-mip6-application-for-ofh-tokens/10605, September 2021. (Accessed on 12/14/2021).

[Sal] Mike Sall. Introducing unique identity (uid), the first nft for identity. — goldfinch_fi — nov, 2021 — medium. https://medium.com/goldfinch-fi/introducing-unique-identity-uid-the-first-nft-for-identity-830a89207509. (Accessed on 12/16/2021).

[TFC] Truefi v3: Credit model & new asset support — by trusttoken — truefi. https://blog.trusttoken.com/truefi-v3-credit-model-new-asset-support-a7cf73a37270. (Accessed on 12/14/2021).

[TFR] Truefi 2021 protocol roadmap: The future of uncollateralized defi lending — truefi. https://blog.trusttoken.com/the-truefi-protocol-roadmap-92f388ac6eb3. (Accessed on 12/13/2021).

[TFS] Github - trusttoken/truefi-spec: Under-collateralized lending and defi smart contracts. https://github.com/trusttoken/truefi-spec. (Accessed on 12/13/2021).

[Trua] Truefi ceo interview - uncollateralized defi loans! — defi now - youtube. https://www.youtube.com/watch?v=IyvXXe8P1O8&t=0s. (Accessed on 12/13/2021).

[Trub] TrueFi. How truefi works — defi uncollateralized lending. https://youtu.be/i8NoS5zRj9Q.

[Wik21] Wikipedia. History of banking — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=History%20of%20banking&oldid=1054610199, 2021. [Online; accessed 13-December-2021].

[Wor] World bank sme finance: Development news, research, data — world bank. https://www.worldbank.org/en/topic/smefinance. (Accessed on 12/16/2021).

A Example TrueFi Forum Post

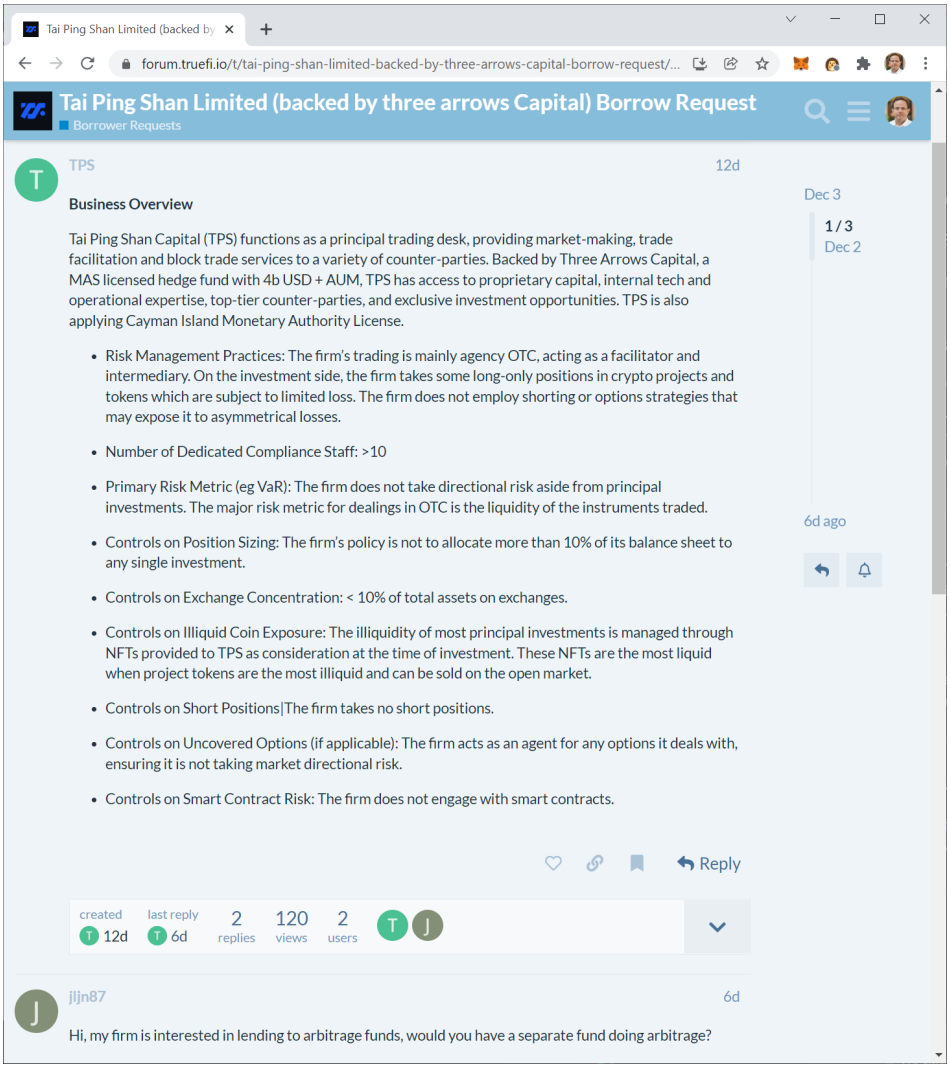

Figure 12: Borrower Introducing Itself on the TrueFi Forum (forum.truefi.io)