LINK Total Return Swap LitePaper

Overview

- A simple, easy to use and understand Total Return Swap smart contract that has similar functions to an Over The Counter (OTC) derivative - see Contract Examples to get straight to the point.
- Chainlink node operators can hedge their risk, ensuring they maintain positive margins during service agreements.
- Speculators can gain leveraged synthetic exposure to the LINK token.
- Using Chainlink Oracles we have secured the data inputs for our contract.
- The exact same contract can be used by a node operator looking to hedge their ETH risk (Synthetically or with wETH).
- This TRS smart contract implementation is nothing ground-breaking, it is intended to be built on top of and serve as an example of what can be built using Ethereum and Chainlink.

Motivation

In order to become a performant node operator on the Chainlink network, one must either own or have access to the LINK token. For high value contracts, large amounts of collateral in LINK may be required. During a service agreement where a node operator must put down collateral, their LINK is fully exposed to market fluctuations and they do not have the ability to withdraw their LINK.

Once the service agreement is complete, an operator's asset may have dramatically depreciated in value, disincentivizing future node operation. By using the Total Return Swap a Chainlink node operator can hedge their exposure to the price fluctuations of LINK by selling their risk to a counterparty, who wants leveraged synthetic exposure to LINK, for an interest rate fee.

The barrier to entry for an oracle to attain LINK can be reduced via staking pools and other lending protocols, and then services such as the LINK TRS may be used by these oracles to hedge their asset risk and earn a stable income through the services that their oracle provide.

Total Return Swap Definition

A Total Return Swap (TRS) is a financial derivative that enables Counterparty A (Alice) to take a long position and gain synthetic exposure to an asset. Counterparty B (Bob) can take a short position, with or without requiring ownership, on the underlying asset of the contract. Alice pays Bob an interest rate that is generally equivalent to the borrowing cost of the underlying asset's notional value. In most cases Bob uses a TRS to earn interest whilst having no exposure to price fluctuations of the underlying asset.

Contract Architecture

The contract architecture displayed below is an implementation of an early MVP launch and thus any future implementations haven't been discussed in the contract architecture. All code is viewable at: https://github.com/securedatalinks/LinkTRS

Initiation of Contract

Step 1: Input Economic Variables

The Maker (Bob) of the contract defines the economic terms of the agreement by inputting the variables (examples in brackets) into the frontend at www.liquidity.link which consist of the:

- Notional Value ($1,000,000 USD of LINK at current price)
- Term (100 days)
- Interest Rate (10% PA)

- Minimum Required Margin - MRM (15% of Notional Value)
- Default Fee Rate (5% of Notional Value if it falls below MRM)
- Defines Acceptance Period (Up to 10 minutes after creation a taker can submit their response, see more in appendix)
- Defines the acceptable threshold of price variation within the acceptance period to initiate a contract (d%=0.25% of deviation from original LINK price)

Submits variables into the contract by depositing an amount of USD (in DAI) above the MRM

Step 2: Acceptance of Contract

The Taker responds within the acceptance period and sets a bid for the Interest Rate they're willing to pay the Maker to gain exposure to the underlying asset:

- Interest Rate Bid (10.1% - greater than the defined interest rate by the Maker)
- Acceptance Period
    - Calculate Notional Value in USD at the creation of contract based on amount of LINK/USD
    - Calculate Notional Value in USD at the end of the end of the acceptance period and use this at the Notional Value of the Contract
- Check if the price of LINK at the beginning of the acceptance period (p_0) has deviated:
    - If change in price > d% then the contract is void
    - If change in price < d% and other conditions hold

The Taker submits the Interest Rate bid into the contract by depositing an amount of USD (in DAI) above the MRM.

Step 3: Selection of Taker

At the end of the acceptance period, if all conditions hold, then the contract is finally initiated and is now live. The NPV calculation execution at the end of the acceptance period can be used as a reference point for a future remargin.

Live Functions

** Executable functions post contract-initiation **

<u>Withdraw and Deposit function</u>

- Taker Margin Account (TMA) and Maker Margin Account (MMA)

- The MMA and TMA allow the Taker and Maker to withdraw and deposit funds into the contract. Funds are exchanged between these accounts by calling the remargin function.
- Funds are deposited into the contract to initiate it and are then stored in this account during the contract
- Funds can be withdrawn or deposited by the Maker and Taker at any point in time with the condition that the account's value exceeds the MRM.

Remargin function

Stage 1:

The Net Present Value (NPV) is calculated and the funds are transferred from the MMA and TMA so that the position of the counterparties is updated in the contract. This contract utilises a running NPV style calculation in order to remain in line with solidity logic. The previous NPV is taken into consideration when calculating the up to date NPV as displayed below in Figure 1.

Taker Receives =

- $((P_n / P_{n-1}) - 1) * \text{Notional Value}$

Maker Receives =

- $((T_n - T_{n-1})/365 * R^*) * \text{Notional Value}$

NPV (Net Present Value) =

- $((P_n / P_{n-1}) - 1) - ((T_n - T_{n-1})/365 * R^*) * \text{Notional Value}$

Where:

- T_n = Time of new remargin
- T_{n-1} = Time of previous remargin
- R^* = Interest rate
- P_n = Price of the asset at new remargin
- P_{n-1} = Price of the asset at previous remargin

The conditions for which an execution of a funds transfer from one account to the other are as follows:

- If the NPV value is **Negative** - Maker receives
- If the NPV value is **Positive** - Taker receives

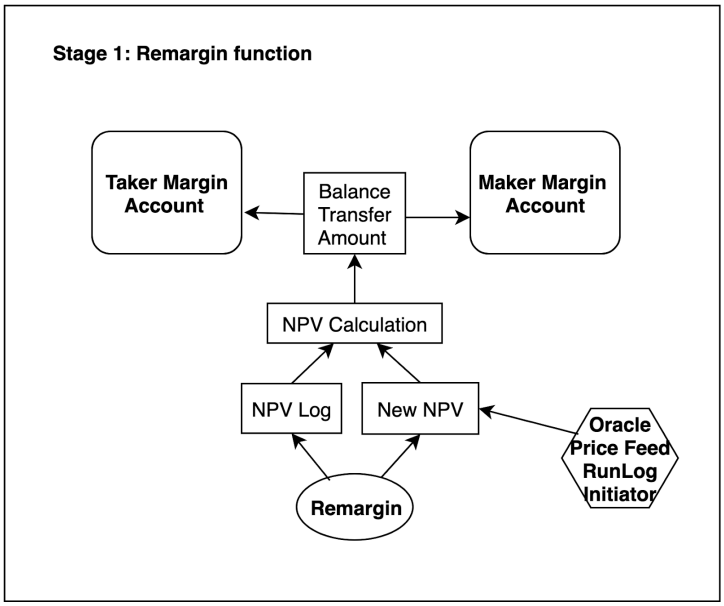


Figure 1: Remargin Call

Stage 2:

Once the remargin balance transfer has been completed the contract checks the TMA and MMA to make sure both remain above the Minimum Required Margin (MRM). If one of the accounts has fallen below the MRM then a default fee is extracted from the defaulting party (agreed upon in the economic terms of the contract). The liquidation process is then triggered and unlocks remaining funds for both counterparties to remove from the contract as displayed in Figure 2.

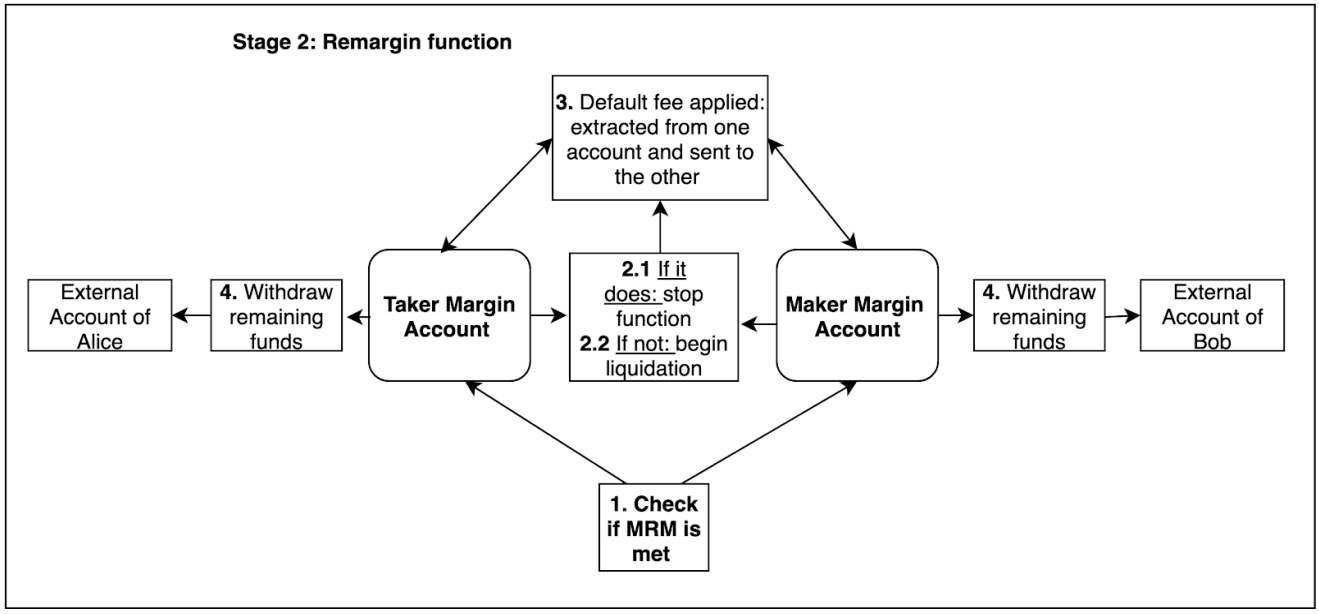


Figure 2: MRM check

Settlement Function

The settlement function will utilise a Chainlink RunAt Initiator (discussed below) to trigger a remargin and NPV calculation at the expiry of the contract if the contract hasn't already been liquidated before expiry. After the settlement function has been executed the funds will be unlocked and the counterparties will be free to withdraw from the contract.

Chainlink Price Oracle Usage

RunAt Initiator

The RunAt initiator enables the Maker to specify a time for an oracle to respond. We use this initiator for:

1. Acceptance Period: At the end of the Acceptance Period get LINK/USD price
2. Expiry of contract: At the end of the contract get LINK/USD price

The reason for the use of the RunAt Initiator is that at the end of the Wire Frame an exact price which is used to calculate the original notional value of the contract and also to verify whether or not the price deviation has been greater than the set threshold. In the expiry case, the NPV must be calculated at the end of the contract and thus an external entity cannot be relied upon to call the settlement function on time.

RunLog Initiator

RunLog Initiator allows an oracle to watch the blockchain for any log events that include the JobID of the initiator. Any event that matches the log event signature of the oracle contract triggers the oracle to parse data and initiate a new log run.

This functionality is used when a remargin call is made as seen in Figure 1 (the oracle) - triggering the oracle to respond with pricing data so that an NPV calculation can be performed. This allows a counterparty to call the remargin function at any time and have very little delay in utilising real-time pricing for the underlying asset price.

For more information on the Chainlink initiator specs: https://docs.chain.link/docs/initiators

Example: Bob Profits

In the scenario Bob profits due to the price of LINK falling. Before the initiation of the contract Bob and Alice utilise the platform to agree on the economic terms of the TRS in Figure 3 below.

Economic Terms of TRS

Notional Value: **$1,000,000** USD of LINK
Interest Rate: **10%** PA
Term: **100** Days
Min. Req. Margin: **15%** of Notional Value
Default Fee: **5**% of Notional Value

Figure 3: Economic Terms

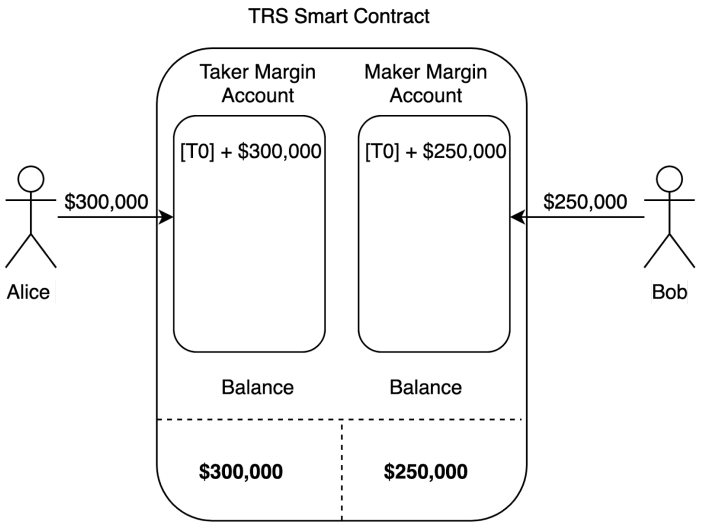


Figure 4: Initiation of Contract

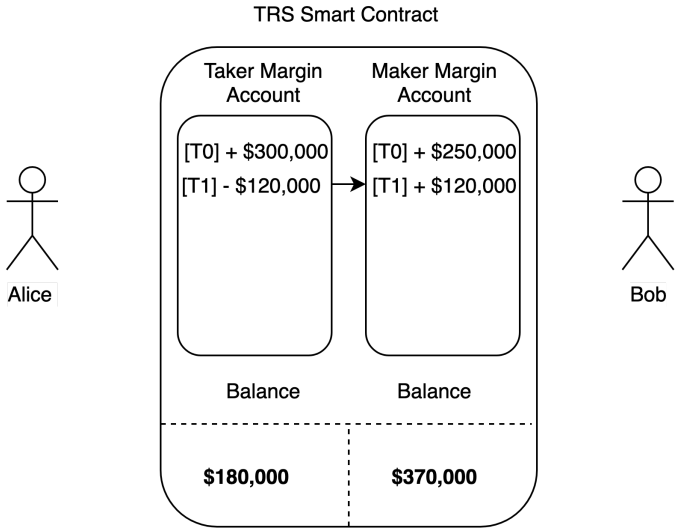


Figure 5: Remargin Call

At T_0 Bob the Maker creates the smart contract, codifying the economic terms and depositing $250,000 USD in the MMA (only requires $150,000 - but in order to remain above the MRM deposits an extra 10% of the notional value to hedge against any volatile price movements is added). Alice responds to the contract by depositing $300,000 into the TMA in Figure 4.

At T_1 in Figure 5 Bob or a 3rd party acting on behalf of Bob calls the remargin function due to a price decrease in LINK price - remargining $120,000 from the TMA to the MMA (adjusting for price and interest payments to date).

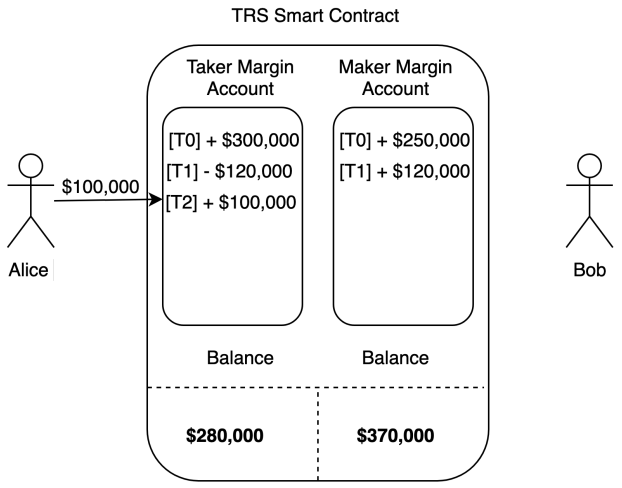


Figure 6: Alice Deposits

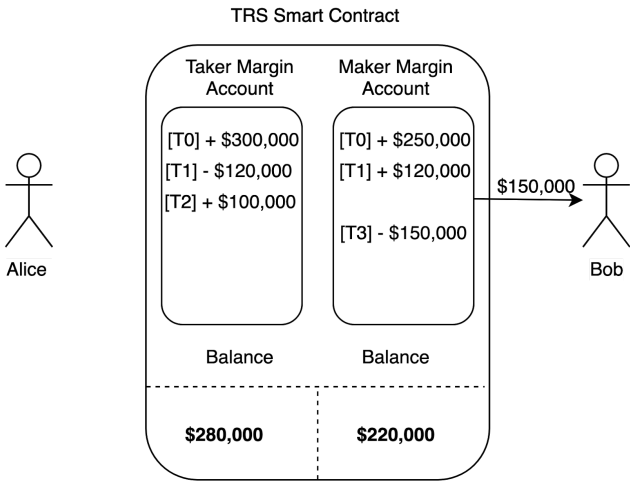


Figure 7: Bob Withdraws

At T_2 after having a balance of $180,000 and being very close to falling below the MRM Alice deposits an extra $100,000 as in Figure 6.

At T_3 after having 37% of the Notional Value of the contract in the MMA bob withdraws $150,000 as in Figure 7.

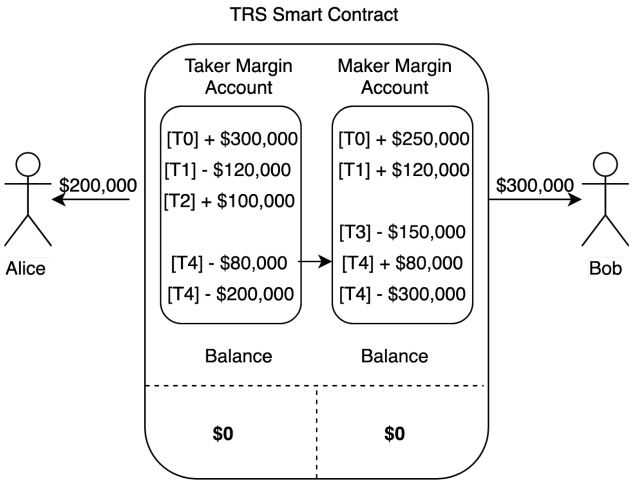
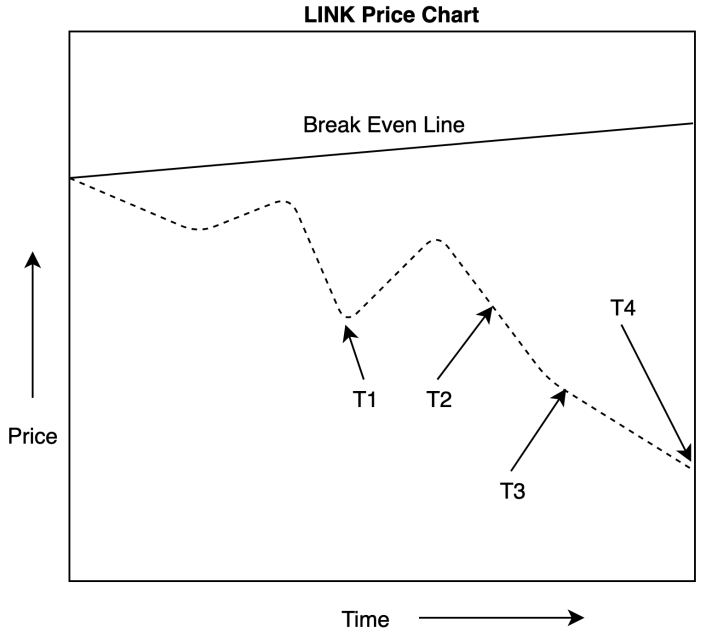


Figure 8: Settlement of Contract

At T_4 in Figure 8 the settlement of the contract is initiated after the contract expires (completing the term). As illustrated in Graph 1 below, the price of LINK has again fallen since the last remargin. In finalising the accounts, $80,000 is remargined into the MMA from the TMA. Alice and Bob can then withdraw from the TMA and MMA respectively, leaving both accounts at $0. Bob has attained $200,000 from the trade and Alice has lost $200,000. The market value of the LINK has dropped from $1,000,000 to $827,397.26 and thus if Bob held the underlying asset he would have effectively mitigated any asset depreciation and earned $27,397.26 USD in interest.



Graph 1: Price Chart

Illustrated in Graph 1 are the times the ***state changes*** occur in the smart contract. The break even line is on an upward slope due to the interest rate payments that Alice would be making to Bob and consequently the interest payments Bob would be making to a Bank if he had to get a loan to purchase the underlying asset.

Example: Bob Defaults

The exact same economic terms used in Figure 1 are used again for this contract.

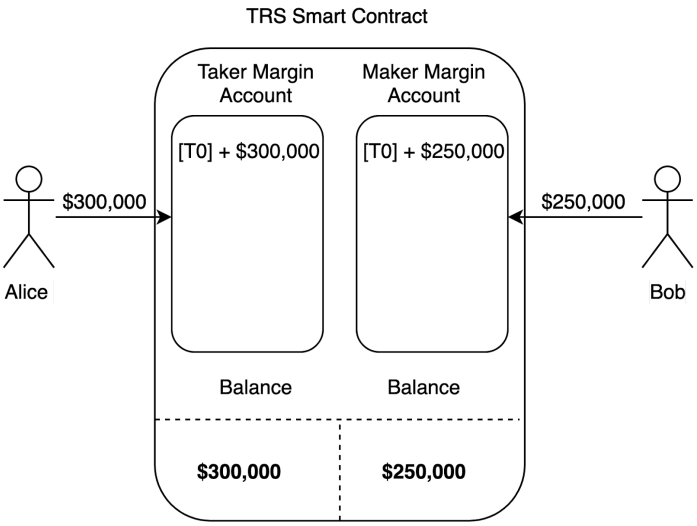


Figure 4 & 9: Initiation of Contract

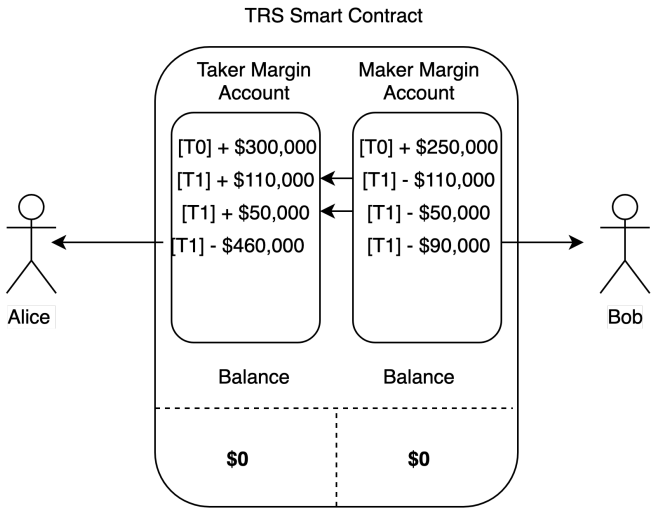


Figure 10: Default & Settlement

The initiation of the contract is replicated from the first example (Figure 4 & 9) and then the price of LINK rises to where Bobs funds in the MMA fall below the MRM.

At T_1 Alice calls the remargin function and $110,000 is remargined from MMA to TMA (Figure 10). The function then checks the balances held within both the MMA and TMA and finds that Bob is $10,000 short of the MRM. This triggers the default fee payment of $50,000 (5% of Notional Value) which is paid into from the MMA into the TMA. The contract is also terminated.

Both parties can then withdraw their funds from the TMA and MMA. Bob loses $160,000 and Alice profits $160,000. Bob's net position is -$50,000 (default fee) plus the interest payments that were made into the MMA account from the Taker.

Design Considerations

Contracts For Difference (CFD) & Synthetic Exposure

Some existing Defi protocols allow an entity to gain leveraged exposure to an asset without having to own that asset.

- There are several different approaches to this form of margin funding on Ethereum which utilise CFD style architectures. Market Protocol [1], dYdX [2] and bZx [3] are successful examples.

- The UMA protocol [4] takes a similar approach but instead utilises the TRS contract format - calculating the NPV as the price changes and then updating accounts based on their position in accordance to the economics of the contract.

Rehypothecation

In current traditional legacy markets, traders are able to use the same underlying collateral to secure multiple different positions, effectively increasing their leverage x-fold. In some extreme circumstances, parties can leverage themselves more than 70:1. Rehypothecation in legacy markets is available because traders and entities that take out these positions have legal reputations and legally enforceable repayment obligations. With rehypothecation being used, circumstances of large fluctuations in rates or prices means that traders may not have the means to repay defaults, increasing the overall market risk.

In the pseudonymous crypto-space where users don't know who they are trading with, out of necessity Defi projects have taken risk-averse approaches that require proof of collateral. Funds are held in escrow when collateral is required and those funds cannot be used in any other smart contract without breaking solidity logic. There are rehypothecation opportunities on Ethereum such as Compound Finance's lending protocol [5], as well as the theoretical Yield

Protocol [6] (implemented by UMA recently) that enable holders of a minted token to earn interest on the underlying locked token.

Compound Finance has implemented a mechanism to incentivize borrowers and lenders to use their platform by having a perpetual agreement which leverages a Supply & Demand model to generate their Annual Percentage Rate (APR) for all borrowers and lenders. When minting cTokens an amount of the original Asset is deposited into Compounds smart contract and in return an amount of cTokens is minted which the user then holds. As time progresses you earn the APR without increasing the amount of tokens you hold in your wallet.

By using the Yield protocol, similar to the Compound protocol, a party can purchase $100 of yToken (e.g. yUSD) at an example discounted price of $0.97 and receive 103.09 yUSD. After holding on to them for a period of time, these 103.09 yUSD are redeemable for $103.09 of an underlying asset that is locked in the Yield Protocol at expiration of the contract. As with Compound's cUSD, yUSD can be kept in a separate margin account and be earning interest.

UMA Protocol

In the LINK TRS contract, the UMA on-chain TRS design has been utilised and extended. To further explain the UMA protocol, it proposes an interesting Data Verification Mechanism (DVM) [7] that in theory acts as a second layer, 'last resort', oracle mechanism that can be used when one of the counterparties in a contract wishes to dispute an oracle's price data. The DVM can work in tandem with Chainlink being primary data feed on-chain and the DVM being the dispute mechanism.

One tradeoff to using the DVM is that all contracts must use a bytecode readable contract format that is compatible with the DVM system. The system then administers fees which determined from reading the bytecode - fees are used to ensure that of corrupting a contract (CoC) is higher than the profit from corrupting a contract (PfC). This standardised form of contract may reduce the confidentiality and the flexibility of agreements.

DVM and other layer two dispute mechanisms such as Chaugur [8] can undermine the deterministic qualities of a smart contract, as they require counterparties to retain their collateral in their margin accounts for a certain period post contract execution. The Chainlink team expects exceptional security at scale once reputation, staking, and threshold signatures are live on Mainnet. But, for high value contracts, an option to dispute may be in high demand and may also be one solution to the bottleneck for Defi adoption as a whole.

Gas Price

Automated remargining on-chain would be a prohibitively gas expensive feature. Therefore for our contract we have opted to build a mechanism that allows the user to call the remargin function or have a 3rd party call the remargin function whenever it is economically efficient to do

so. This places the onus on the user to organise the calling of the function - yet saves massively on transaction fees.

For example, one remargin function call through metamask requires 0.0076 ETH (~$1.35 USD) for the slowest processing times and up to 0.076 ETH (~13.8 USD) for faster processing times. By limiting the number of remargin calls made to our contract, we are able to limit expenditure.

Potential Risks

Counterparty risk

Taker & Maker: There is a small amount of counterparty risk observed when either Bob or Alice aren't triggering the remargin call when needed.

> *For example:* Alice and Bob both don't check the contract status for an unspecified amount of time and in that time Alice has fallen below the MRM and also the Default Rate Penalty. Bob finally remargins the account yet Alice doesn't have enough left in her account to pay the default rate or in the worst case the updated NPV.

Maker: The Maker brings counterparty risk to the contract by the Makers exposure to the risk of losing his LINK holdings if he fails to provide data in a correct manner or on time. Thus in the case of a loss on the Penalty Deposits function, Bob is essentially in what is called a position of *naked short selling* - and thus his losses are unlimited at this point. The likelihood of Bob to wish to terminate his position is high and thus he may default on the contract, Alice no longer having the upside of being exposed to the underlying asset.

Systemic risk

There is the reliance placed on the security and stability of DAI as the underlying reference asset that is being lent out on the Compound Finance protocol.

Limitations

Low Liquidity

The chicken before the egg idea is prevalent for this use case of a TRS when it comes to liquidity and transaction volume. Chainlink Network usage is dependent on a multitude of factors. There may be a small amount of oracles who demand to be the Maker in these early market stages which are met by an unbalanced high demand from Takers. With attractive

enough interest rates offered by takers (LINK token speculators), more oracles may be tempted to utilise the product.

As is the case with the UMA protocol this contract format certainly won't be limited to a single LINK TRS market. For other proof of stake coins and cryptocurrencies, TRS products may become increasingly attractive and new TRS markets may emerge. OTC contracts can also be created and executed using this protocol with very little change to the core architecture. There would be multitudes of possibilities for a Defi platform where contract customisation is a key feature.

Not Utilising Bob's Penalty Deposit LINK

Representing the LINK that Bob has, which is locked up in a service agreement as a penalty deposit, in his margin account would be a desirable feature. It would mean that Bob would only require ownership of the LINK and not have to provide extra liquidity for his margin account, not needing to go to the cDAI or DAI markets.

Future work on this matter could be applied to formulate a method in which the funds collateralized in a service agreement can be utilised in the TRS.

Future Work

Further Utilisation of Chainlink Oracles

In instances of contract default and liquidation, currently if either party falls below the MRM the contract is not automatically executed. In future we would use Chainlink remote initiators to monitor the contract, and automatically execute a default function if either party falls below their margin requirement. As well as in the case of a default, remote initiators can be used to automatically remargin accounts based on proportional contract value changes - e.g. IF the contract moves more than 2%, THEN execute remargin function.

Automated scheduled remarging (e.g. once per minute) could also be desirable, although there would be high fees associated.Fees for automated remargining could be determined at the beginning of a contract. These fees could be evenly split upfront be each party, or incremental fees could be taken from each margin account at the time or remargining.

The MRM may be updated by an Oracle based on a change in market volatility rates. Each party could agree upon the use of an oracle that updates the specified MRM based on a change in market indicators.

Reputation

Future iterations of such a platform could provide Taker and Maker address profiles with associated contract histories. Displaying a counterparty's past trading history would allow for their reputation as a trader to be determined. Information such default rates can be clearly displayed. Reputation information makes for more informed decisions on behalf of the trader as to the counterparties they wish to engage in contracts with. Certain blacklists can be established for traders that have high default rates.

The node operator's Oracle reputation can also be displayed and used for determining staking and lending rates to an oracle. In the scenario where Bob's LINK that is locked up in a SA is used as his TRS margin, reputation can determine the likelihood that an oracle will incur a penalty payment and lose their LINK that was securing the TRS.

Interest-bearing Tokens in Margin Accounts

Utilising the rehypothecation opportunity that the Compound Finance protocol provides, all collateral deposited in margin accounts can earn interest or have a stablecoin token redeemable for a certain amount of interest. Similar tokens such as the yToken can also be used as collateral.

Termination Function

A termination function could also be added and whoever initiates it pays a fee if both parties choose to accept termination of the contract (the fee being less expensive than the default rate). This fee can be agreed upon in the terms before a contract's initiation.

Auction During Acceptance Period

Once the contract variables are specified (MRM, notional value, term, acceptance period, etc.), the contract can enter an auction phase where Takers compete to be chosen for the TRS. The Maker may set a minimum required R*, and then Takers can choose to bid on R*. The auction would allow the Maker to decide on which Taker he would like to choose based on the respective R* bids and the reputation of the Taker.

If there are no bids during the period, then the Maker will have to recreate their contract and reopen the market. At the end of the acceptance period, if the percentage of notional value deviation is greater than the prespecified deviation limit then the contract is voided.

Exchange Solution

A design consideration for a future implementation of the LINK TRS platform would be to embed the Uniswap exchange [9] (assuming the liquidity on the LINK/ETH pairing is large enough) so

that Takers and Makers could purchase whichever underlying asset is required for their contract or desired post contract resolution.

Oracle Selection

It is within the counterparties interest to have the power to choose the oracles they will use for their single agtreement. If users would like to pay for a greater quantity of high premium, high reputation oracles then they may do so. The cost for increased security could be paid to the platform as a fee. The platform could even then automatically use an exchange like Uniswap, utilising the contract fees to acquire LINK and compensate the premium oracles.

Conclusion

The LINK TRS contract using the Chainlink network's price oracles is a quick and crude hack of an easy to use smart contract that can be valuable for the Chainlink node operator. The Chainlink network is an integral element of LINK TRS, adding security and robustness through its decentralised, economically secure oracle middleware.

The TRS contract itself can be used in any market where there is a secure data feed for the price. Outside of the crypto space for example in the stock market, the TRS can be used for assets such as AAPL stock or commodities like Gold. Decentralised finance opens up financial markets and opportunities that have previously had extremely high barriers to entry. This space should continue to be explored so that valuable, transparent and unstoppable deals can continue to be made.

Acknowledgements

Acknowledgments must go to the team at UMA for illustrating how a TRS could be implemented and used on the Ethereum and the Chainlink team for creating such a developer friendly environment to utilise external data in Ethereum smart contracts - successfully enabling a trustless state change in two entities funds in a derivative contract.

Reference List

[1] - Market Protocol, https://marketprotocol.io/

[2] - dYdX Exchange, https://dydx.exchange/

[3] - bZx Protocol, https://bzx.network/

[4] - UMA Paper, https://github.com/UMAprotocol/whitepaper/blob/master/UMA-whitepaper.pdf

[5] - Compound Finance, https://compound.finance/

[6] - Yield Protocol, http://research.paradigm.xyz/Yield.pdf

[7] - UMA's DVM paper, https://github.com/UMAprotocol/whitepaper/blob/master/UMA-DVM-oracle-whitepaper.pdf

[8] - Chaugur, https://blog.chain.link/showcasing-the-winners-of-the-etherlin-zwei-hackathon/

[9] - Uniswap Exchange, https://uniswap.exchange/swap

Appendix

Service Agreement

The Chainlink Service Agreement is a contract that a job requester uses to engage oracles for jobs. The terms of the agreement are first specified off-chain by the job requester. Within the terms include certain encumbrance parameters (payment terms, penalty deposit term, minimum reputation, number of oracles etc.) as well as job requirements and job specifications. Once these terms are specified, the terms of the SA are delivered on-chain where oracle either accept or decline the agreement. The LINK TRS allows those oracles engaged in a service agreement, which requires requires LINK liquidity, to hedge their asset risk.

Penalty Deposits

As mentioned above, within the service agreement a job requester can specify the value of a penalty deposit that an oracle must put down as collateral to ensure that they perform honestly and well. Carol will be able to specify the amount and frequency of penalty deposits that she requires. In the event of an oracle not responding or answering incorrectly to a request, their collateral is lost.

> Example:
>
> Carol chooses 3 oracles to provide the price of Gold at 11:59pm Oct 27 for a binary options contract where Ben and Jerry (counterparties) have both input $1m USD and therefore stand to lose or win $1m USD pending the result of whether the gold is above or below a certain price they have agreed upon.
>
> Within the service agreement Carol specifies that the minimum required deposit from the 3 oracles will be >50% of the value of the contracts outcome - $500,001. To corrupt 2/3 of the oracles it's going to cost $1,000,002 (economically speaking it isn't viable at this point to corrupt the contract).