{{PegLoan}}: A trust-minimized collateral-backed stablecoin system

Technical research whitepaper v0.1 [changing]

> Notice: This document is a research proposal, and in no way makes promises about the full functionality of any derived implementation, nor does it represent a commitment or intent to build, deploy or operate such a system. Its sole aims is to envision the process of defining, planning and productizing such a system and product.

> Note: Names inside {{}} denote placeholder product names.

Abstract

This research proposal describes a **collateral-backed stablecoin** system implemented on a public blockchain, where the exclusive backing collateral is the **native blockchain asset token**, and where the system is **immutable** and uses a **decentralized oracle mechanism**. Our proof of concept is implemented on Ethereum using Solidity, uses ETH exclusively for collateralization, uses a peg to $USD as its reference currency.

The collateral backed stablecoin systems ensures the stable value of a digital token, through securing collateral of equal or higher value, and by balancing token supply and demand as a response to market conditions, by adjusting monetary variables (eg fees and interest rates). Variations of such a system (eg MakerDAO's DAI), are already offered on public blockchains, however our proposed system contains **no on-chain governance process** and **no tokens for governance or equity**. The proposal posits that eliminating these, in favor of an on-chain incentive system, **reduces centralization**, and **increases the capital efficiency**.

Below we describe the system actors, structure, and operations at various levels of detail, along with some new concepts. A proof of concept implementation of this idea is also available, written in Solidity.

Actors

- **Money users** - These are *regular consumers* that *use*, or *store* the pegged currency money in their blockchain-based *savings account*. The notable target groups that could most benefit from such an offering are *the un-banked* and those with limited access to a *stable Store of Value* (SoV) from countries with dysfunctional monetary policies. This could also be used by the *early adopter community* of a public blockchain.
- **Loan takers** - Most commonly, existing *native token holders* who have decided to *take a loan* against their holdings. The main incentive for this group consists of being *long native token*, while being able to *deploy its value* in other profitable transactions. They may also initially be *motivated by the technology* itself, however this is not a long-term sustainable incentive.
- **Price feed providers** - maintain the system by *providing accurate rates* of exchange between naive currency (ie ETH), the reference currency (ie USD) as well as the pegged currency. These providers are expected to compete in *establishing trust* amongst the loan takers, and in return be *monetarily compensated* based on trust level, as well as *avoiding penalties* by adhering to the loan system rules.
- **Loan liquidators** - maintain the system while *competing* with each other to *liquidate* undercollateralized loans. They monitor loans and trigger liquidation while *speculating* on the future collateralization levels of

target loans as judged by the system.

System operation

Trusted price feed list

The system maintains a list of medium (and high) trusted price feeds, sorted and weighted by both a price feed's revenue pool as well as its issuance allocation. This formula ensures newcomer price providers cannot exert undue influence by instantaneous deployment of capital, while also making sure that loan taker choices in allocating issuance to feeds of their choice also affects the ranking.

The trust rank of each given price feed will be assessed by the system using a custom defined metric calculated by the following formula = `total issuance allocation` x (`revenue pool balance` + `average revenue pool balance`)

The formula is chosen to ensure the ranking maximizes security of the system considering the notable potential attacks that can harm stable operation of the system. The formulation has the following properties:

- **Loan allocation** - Loan takers will be able to directly affect this metric by changing allocations. For example `50%` reduction in allocation will result in a `50%` reduction in the metric.
- **Long-term revenue** - The amount of previous service to the system accounts for a significant part of the metric. This is to prevent potential easy system capture by whales. A top price feed with `2x` the average price feed pool balance will have a `3x` advantage over a new feed with pool balance `0` and the same allocation. A top price feed with `2x` the average price feed pool balance will have a `2/3x` advantage over an average feed pool balance with the same allocation.
- **New entrant competitiveness** - A new entrant should be able to compete and not have an unsurmountable impediment to raising in the ranks. An average revenue pool balance value represents `2x` the metric value of a brand new price feed with the same allocation but `0` revenue pool. That is if the new entrant secures `2x` the allocation of average feeds, it will be ranked above them.

Delayed price aggregation process

The aggregate delayed price is calculated each time the daily delayed prices are finalized, based on a process that aims to provide an accurate daily median price, while reducing or eliminating the chance of system abuse. The system takes the following steps:

- It starts with prices reported by the `20` medium but not high trust price feeds, as reflected by the trusted fee list being finalized
- The process eliminates outliers in order to keep up to `10` reported delayed prices
- Then it adds the `5` high trust price feeds back into the list and eliminates outliers again to reach up to `13` prices
- If the range of prices exceeds `25%` the delayed price sub-system is designated to be in `Dispute` state, which postpones liquidation finalizations and halts instant loans by reducing the instant loan limit rate to `0%`.
- If the range of prices exceeds `5%` state is set to `Unstable` and instant loan limit rate is set to `0.5%`.
- Otherwise the sub-system is in `Stable` state.
- The finalized delayed price is set to the average of all remaining up to `13` reported prices, weighted by each feed's total allocation.

- If in `Dispute` state, all price feeds pay a dispute penalty calculated by the formula: `Feed dispute penalty rate` = `100%` x abs(`Feed price` - `Aggregate price`) / `Aggregate price` / `2`
- If in `Unstable` state, all price feeds pay a dispute penalty calculated by the formula: `Feed instability penalty rate` = `100%` x abs(`Feed price` - `Aggregate price`) / `Aggregate price` / `7`

Instant price aggregation process

The aggregate instant price is calculated each time an instant price is reported, based on a process that aims to enable instant loan taking capability, while reducing or eliminating the chance of system abuse. The system takes the following steps:

- It starts with prices reported by the `5` *high trust price feeds* as reflected by the last finalized trusted fee list.
- The process eliminates any values that are not reported within the last `1 day`, then it eliminates outliers in order to keep up to `3` reported instant prices.
- If the range of prices exceeds `20%` the instant price sub-system is designated to be in `Dispute` state, which halts instant loans by reducing the instant loan limit rate to `0%`.
- If the range of prices exceeds `5%` state is set to `Unstable` and instant loan limit rate is set to `0.5%`.
- Otherwise the sub-system is in `Stable` state, instant loan limit rate is set to `5%`..
- The instant price is set to be the lowest out of the up to `3` remaining values, in order to further protect against system leaking pegged currency issuance

Automated monetary system

The main goal of the automated monetary system is to balance supply of pegged currency with its demand, and maintain a price pegged to the reference currency, as reported by the price feed providers as historical prices. The main variables that the monetary system can affect are the following:

- Price feed revenue rate
- Loan fee rate
- Savings interest rate
- Loan collateral threshold ratio

Peg equilibrium metric - is a measure of how stable the peg is relative to the desired `1.00000` rate, and determines the level of market's stability, overdemand, or oversupply. The metric is calculated as follows: `Peg equilibrium metric` = (`+` or `–`) Σ abs(`Peg price` - `1.00`), where the sum is taken for the days in a row that the peg has been above or below `1.00`, and where `+` denotes being above, and `–` denotes being below `1.00`.

Loan fee rate - Is a yearly rate set by the automated monetary system that determines the fee paid by loan takers upon structural changes made to their loan, such as allocation or issuance changes. For example, at a given point in time, the loan fee rate could be `4%` per year (equivalent to `0.01096%` per day). Assuming no future change to the rate (which is unlikely), the loan taker should project to pay `$400` per year in fees on a loan that has issued `$10,000` worth of pegged currency, upon closing the loan. The actual annual rate will be calculated by adding all the variable daily rates.

The monetary policy engine aims to vary the rates based on the global equilibrium price of the pegged currency, in order to incentivize increase or decrease the pegged currency supply based on the creation or cloning of debt positions, and thus affect the equilibrium price of the pegged currency itself from the supply side.

The daily loan fee rate varies by `1% weekly` towards the *loan fee target rate*.

Base loan fee rate - is the loan fee rate used at the most stable conditions of system operations. It can be set using one of the following strategies:

- Fixed `base loan fee rate` = `2%` (current implementation).
- A fee that very gradually moves up or down (by a maximum of `1% per year`), based on ongoing feed provider voting as part of price reporting process.
- A fee that varies based on past performance of the loan fee rate, selected from past periods where the *peg equilibrium metric* has had the lowest average magnitude, and when *total issuance* has grown the most.

Loan fee target rate - is set by the peg currency supply demand equilibrium policy as the minimum of the fixed value `20%` and the formula: `Loan fee target rate` = `base loan fee rate` - `peg equilibrium metric` x `10` x `1%`. This means a full week of peg price at `$1.02`, will lower the *loan fee target rate* by `1.4%` and a full week of peg price at `$1.05` will lower it by `2.8%`.

Price feed revenue rate - Is a weekly rate set by the automated monetary system that determines what percentage of total system issuance is expected to eventually be paid out to price feed providers. The revenue comes from part of the loan fee stream and so the price feed revenue daily rate has to always be lower than the loan fee daily rate. The rest of the loan fees will go into the savings pool to be paid out to savings account owners. For example, at a given point in time, the price feed revenue rate could be `0.01917%` per week (equivalent to `1%` per year). Assuming no future change to the rate (which is unlikely), a price feed provider can project its revenue pool to receive `$20,000` yearly assuming a total issuance allocation of `$2,000,000`. This could be the case if total system issuance is `$10,000,000` and the providers average weighted allocation percentage is `20%`.

The price feed rate can be set using the following strategies:

- Fixed `price feed revenue rate` = `1%` (current implementation).
- The price feed rate will vary logarithmically based on **total issuance** of the system. See table below for rates corresponding to total issuance.

Total issuance	Price feed revenue rate	Total revenue	Average revenue
$1	2.5%	$0.25	$0.01
$1,000	2%	$20	$0.80
$1,000,000	1.5%	$15,000	$600
$1,000,000,000	1%	$10,000,000	$400,000
$1,000,000,000,000	0.5%	$5,000,000,000	$200,000,000

Savings interest rate - Is a daily rate set by the automated monetary system that determines the interest paid to owners of savings accounts. For example, at a given point in time, the savings interest rate can be `0.01096%` per day (equivalent to `4%` per year). Assuming no future change to the rate (which is unlikely), the savings account owner can project an interest of `$400` per year on a savings account balance of `$10,000`.

The actual savings interest rate moves by a maximum of `0.5% per week` towards the target savings interest rate set by the automated monetary system. The interest rate target in turn is set to minimum of the two following values:

1. **Target savings interest rate** - set as part of peg supply demand equilibrium policy, which moves in the same direction as the loan fee rate. This value is calculated with the following formula: `Target savings interest rate` = 2 x (`Loan fee target rate` - `price feed revenue rate`). This means at the most stable operation conditions where *loan fee target rate* is `2%` and *price feed revenue rate* is `1%`, the *Monetary target savings interest rate* will be `2%`.
2. **Max feasible savings interest rate** - calculated based on the formula: `Max feasible savings interest yearly rate` = `100%` x `4 quarters / year` x `Total savings pool balance` / `Total savings registered`. The savings pool will hold enough liquidity to cover `1 quarter` of accumulated interest at any time, this is to keep its position at a safe level in case:
    1. Savings pool inflow decreases considerably.
    2. There is a run on accumulating interest in existing savings accounts.
    3. There is a considerable increase in new savings registered.

Loan collateral threshold ratio - Determines the safe level of pegged currency issuance given the backing native token collateral. This ratio can take on of the following forms:

- Fixed ratio of `150%` (current implementation)
- Variable ratio moving at a rate of `1% per week` based on a target rate ranging between `150%` and `120%`, chosen based on changing variability of the native token price, and time since the last destabilizing change.
    - `150%` - 0 days since the last change exceeding `20% per day` or `50% per week` in magnitude, or the system's first day, whichever comes first.
    - `120%` - `5+` years after last change exceeding `20% per day` or `50% per week` in magnitude, or the system's first day, whichever comes first.

System structure

Components

The {{PegLoan}} stablecoin system consists of the following major components:

- Main system
    - Price feeds functional area
    - Monetary policy functional area
    - Loans functional area
    - Savings functional area
    - Liquidation functional area
- Pegged token (aka stablecoin)
- Loan accounts
- Savings accounts
- Price feed accounts
- Liquidator accounts

As part of this proposal, we will discuss in detail the functionality of each component as well as how end-to-end scenarios function.

State

Main system contracts hold the following variables related to the areas of the system like Savings, Loans, and Price Feeds:

- Loan contracts
- Price feed contracts
- Savings account contracts
- Total and individual price feed allocations (in pegged currency)
- Price feed revenue pools (in pegged currency)
- Savings pool (in pegged currency)
- Current (processing) medium trust price feeds
- Finalized (finalized) medium trust price feeds
- Archive of values
    - Frequencies: 7 x daily, 5 x weekly, 5 x 5 weekly (~ monthly), 5 x 25 weekly (~ bi-yearly), 5 x 125 weekly (~2.4 yearly), 5 x 625 weekly (~ 12 years), 5 x 3125 weekly (~60 years)
    - Savings interest rate
    - Loan fee rate
    - Native token price
    - Peg equilibrium metric
    - Total issuance

Delayed price reporting state

The system has two operational areas, the major one relies on **delayed price reporting**, while the other smaller area relies on **instant price reporting**. The area relying on delayed price reporting has the following daily states:

- **Processing** - This state is before a given day's price reporting is finalized. We transition from processing to finalized within `1 day`.
- **Empty** - This is the initial finalized state when no price providers have yet registered with the system. It is very much a temporary and short lived state.
- **Stable**: where no anomalies in delayed price reporting has been detected, and where daily and instant prices are within known variability thresholds `5%`.
    - Currency issuance will be instant within transaction, up to a limit of `5%` total issuance.
    - Loan liquidation will take `1 week`
- **Unstable**: where minor anomalies have been detected such as ~~1. change in instant price is possibly not reported by minority ~~ ~~2. drastic recent changes in allocations such as `10%` drops 3. Volatility of reported prices exceeding `20% weekly` despite lack of dispute.~~
    - Currency issuance will be instant within transaction, up to a limit of `0.5%` total issuance.
    - Loan liquidation will take `1 week` and will be evaluated against all historical native token prices of that week
- **Dispute**: where major disagreement is occurring between medium trust providers
    - Currency issuance will be disabled
    - Loan liquidation will be postponed for the duration of dispute state

Instant price reporting state

The instant price reporting area has separate yet similar states. The state results applies only to currency issuance and is applied on top of the finalized delayed states from `2 days` ago. This means currency issuance will be restricted based on the maximum of each area's state. For example, `Unstable` and `Stable` will result in `Unstable`. Following are the instant price reporting states:

- **Empty** - This is the initial state when no price providers have reported instant prices within the last `1 day`
- **Stable**: where no anomalies in instant price reporting has been detected, and where instant prices are within known variability thresholds `5%`.
    - Currency issuance will be instant within transaction, up to a limit of `5%` total issuance.
- **Unstable**: where minor anomalies have been detected such as
    - Currency issuance will be instant within transaction, up to a limit of `0.5%` total issuance.
- **Dispute**: where major disagreement is occurring between medium trust providers
    - Currency issuance will be disabled

Tokens and currencies

The system supports pegging to any relatively stable real world currency, as well as other relatively stable baskets of assets, always backed with the most trust-minimized blockchain collateral (ETH in case of Ethereum, BTC for Bitcoin). The main viable implementation of this system however will focus on the US dollar due to its relative ubiquity at time of this writing.

Pegged currency (money)

The pegged currency is the money product that is ultimate offered to everyday digital money users. It will have a stable value, the unit of which is widely accepted and used in commercial transactions by buyers and merchants. The Ethereum based proof of concept implementation of this system will use the ERC20 token standard to represents the value of the US dollar ($USD).

The token requires the base functionality already available in common programmable digital tokens, such as basic transfer functionality between accounts. The Ethereum community's version of such functionality is described by the official ERC20 standard(also described here). Additional functionality should be implemented to respectively **mint** or **burn** tokens upon the pegged currency's **issuance** or **return**.

Reference currency

This is the real world currency, or unit of account, the pegged currency will be pegged to. For example, the US dollar ($USD) is the most commonly used reference currency in most existing stablecoin implementations.

The system is designed such that it would work with any relatively stable currency, or basket of goods, assets and/or currencies. For example the Euro can be a reference currency, and so could be a basket of consumer goods defined by an international body. Any blockchain user would theoretically be able to create a new currency peg to their target currency reference, by deploying the open source contracts.

Backing asset token

As mentioned, any public blockchain's native asset token satisfies the role of backing collateral asset, given that it is likely to be:

- The most third-party trust-minimized token on that blockchain.
- Widely used as a Store of Value (SoV) by the blockchain's early adopters.

It is essential for the system to prevent significant issuance of pegged currency, without securing a corresponding value of backing asset, per loan instance, and for the system as a whole. Similarly, it is crucial for the system to respond to backing asset release requests, when corresponding pegged currency is being returned to the system. In a healthy system, the value of the backing asset token is greater than the pegged currency by a healthy margin, in order to insure against the possibility of a major devaluation in that backing asset.

Loans

Loans (aka debt positions) are the logical structures that hold the state and value of the collateral backing aspect of the system. They have the following notable properties:

- There is a separate loan contract instance per instance of actual loan taken
- The loan contract holds the state and value related to each loan
- The loan taker is the ultimate owner of the value contained on a loan instance, and directly interacts with it.
- The greater system is consulted every time a change is made to the loan. Changes disallowed by the main system, result in automatically reverted transactions.
- A specific threshold of backing to issuance is considered healthy and enforced by the system. For example at one point in time: `backing / issuance = 150%`.

Loans consist of the following state or calculated values:

- **Native token deposited** into the contract.
- **Pegged currency issued** by the loan taker.
- **Pegged currency stored** in the loan.
- **Fees accumulated** since last structural change (Calculated).
- **Liquidation requests** and state.

Currency issuance

Currency issuance is the process of issuing new pegged currency tokens that are sufficiently backed by the native token collateral held in a loan. The loan taker requests a specific change in pegged currency issuance. The system determines sufficient backing by comparing a calculated leverage ratio with a threshold rate (typically `150%`). The loan leverage ratio is calculated as follows: `Loan leverage ratio` = `100%` x `native token deposited` x `native token price` / `total pegged currency issuance`.

For instant issuance purposes, the native token price is determined in the following sequence:

- If the instant price reporting system is not in `Empty` or `Dispute` state, the instant price is used
- Otherwise, if the last finalized day in delayed price reporting system is not in `Empty` or `Dispute` state, the last finalized delayed price is used.
- Otherwise, the price is calculated by taking the weighted average of latest prices reported by allocated price feeds.

Daily issuance limit - There will be a daily currency issuance limit of `5%` as a percentage of the total issuance balance to date. The limit is in place due to the potential for bad actors to abuse the system by taking instant loans and issuing currency during large price drops (see Instant issuance before reporting large price drop). This specific value of the limit would cap any potential large scale abuse, but would also minimize the adverse effects of loan usability and potential capping on the system's issuance growth.

This daily issuance limit is reduced to `0.5%` in case the system foundation goes into `Unstable` state. The limit goes to `0%` in `Dispute` state.

Loan fee

Loan takers pay a fee based on the value of the loan's issued pegged currency, and based on a variable daily percent rate (See ***Loan fee rate*** under monetary variables) set by the main system's monetary policy engine. The fee has to be paid as part of structural changes made to the loan, including changes to allocation or issuance. For example, at a given point, the loan fee rate might be `4%` per year, (equivalent to `0.01096%` per day). Assuming no change to the rate (unlikely), the loan taker should expect to pay `$400` per year in fees on a loan that has issued `$10,000` worth of pegged currency, upon closing the loan. In reality, the yearly fee is determined by the sum of all daily rates throughout that year.

A part of the fees paid by loan takers, will go to the price feed providers. The specific value going to price feed providers is calculated based a weekly variable rate determined by the monetary engine (See ***price feed revenue rate*** under monetary variables). This value is allocated to specific price feed providers based on the allocation table values (See loan price feed allocation) of the loan.

The rest of the fees paid by loan takers, will go to the savings pool, to eventually be distributed to savings account owners based on the monetary system's savings interest daily rate (See ***Savings interest rate*** under monetary variables)

~~**Fee prepayment** - Issuing new pegged currency requires pre-paying a certain portion of the fees meant for an upcoming period (`35 days`). This measure is meant to reduce the ease by which a whale attacker can create loans, issue currency, and create artificial issuance allocations for their own rogue price feeds. See Price feed capture by whales~~.

Loan price feed allocation

Every loan (aka debt position) can allocate one or many price feeds providers by weight for their loan. For a given loan, these allocations are weighted by percentages that add up to `100%`. For example, a loan may allocate `40%` to the price feed maintained by *Foundation X*, `35%` to *Rating company Y* and `25%` to *Financial company Z* for a total of `100%`.

The allocations are meant for three overlapping purposes:

1. **Voting mechanism** - Allocation functions as a voting system, that affects the aggregate trust assigned to each feed provider.
2. **Funding mechanism** - Allocation helps determine the extent to which each feed provider will receive funding to maintain a sustainable business model.
3. **Safety fallback against liquidation** - In case of long lasting price reporting disputes in the system, allocation helps the system know what criteria to use for testing each loan's liquidation conditions.

A part of the fees paid by loan takers, will eventually go to the price feed providers as revenue. The specific value going to price feed providers is calculated based a weekly variable rate determined by the monetary engine (See ***price feed revenue rate*** under monetary variables). That value is distributed to specific price feed providers, based on the allocation table values of the loan. Loan takers are responsible for determining the allocation mentioned above, based on which price feed providers they trust and/or want to financially support.

At level of the entire system, we add up the total allocation values resulting from allocation percentages multiplied by the issuance value of each loan. For example consider a single loan with issuance value `$100,000`, that has three allocations A at `50%`, B at `35%`, C at `25%`. The single loan will contribute to the price feed providers' revenue pools by `$50,000` for A, `$35,000` for B and `$25,000` for C. If the whole system consisted of say `200` loans, that are same as the one above (very unlikely), the total allocation numbers for the price feeds will be `$10,000,000` for A, `$7,000,000` for B and `$5,000,000` for C. Based on a projected *price feed revenue rate* of say `1%` per year, the projected increase to each of their revenue pools would be `$100,000` for A, `$70,000` for B and `$50,000` for C.

From the perspective of reducing risk of system capture, it is ideal that there be more price feeds allocations, and that they be easily changeable in case of bad behavior by specific price feeds. On the other hand however, an excessive number of allocations will cause excessive gas cost and cognitive overhead for the loan takers, as well as stakeholders. An average of 3 allocations is a good target, thus a max value of 5 is reasonable. See maximum price feed allocation.

Price feed allocation process should be a separate call to the loan contract, instead of being part of the loan creation transaction. This decision is made despite adding an extra step, and additional complexity, to the process, based on the following reasoning:

- **No single allocation API** - We aim to disincentivize single allocations, and thus choose not to provide an API as a simpler version of contract creation with single allocation.
- **Allocation as a separate step** - We aim to make allocation changes no more difficult than the initial allocation, and so, we will reuse the allocation process both for initial creation and subsequent changes made to allocation.
- **Unfortunate but unavoidable friction point** - The level of complexity proposed does not exceed the loan takers assumed level of sophistication and tolerance for usability costs.

Liquidation process

Liquidation can be proposed by any blockchain user, who agrees to put up a deposit in pegged currency, equivalent to the outstanding issuance. However the final liquidation decision would be made by the system, only if the specific monetary conditions of the position merit it over a period of `7 days` (where the system is not in `Dispute` state). The decision for each given day is made through comparing a leverage threshold rate with the following calculated leverage ratio: `Loan leverage ratio` = `100%` x `native token deposited` x `native token price` / `total pegged currency issuance`

For liquidation to occur, the liquidation conditions must be met over all of the `7 days` mentioned above. If even one day's aggregate system price invalidates liquidation conditions, the liquidation request will not be finalized. Partial liquidations are not allowed. Adjustments to pegged currency deposits are not allowed to be made at any time after request is submitted.

Common liquidation conditions - Liquidator agrees to deposit pegged currency sufficient for paying off the loan issuance, as well as any outstanding fees in pegged currency. In this case the liquidator agrees to pay a deposit calculated as follows: `liquidator deposit` = `loan total issuance` x (`100%` + `reward bid percentage`) + `outstanding fees`. Upon successful finalization of the liquidation request, the liquidator would receive a payment in native token calculated as follows: `liquidation payment` = (`outstanding fees` + `loan total issuance` x `reward bid percentage`) / `native token price on liquidation day`. The loan owner will have control over any remaining native token stored in the loan contract. Upon failed finalization the liquidator is returned the whole deposit.

Undercollateralized liquidation conditions - There is a special case, where a loan collateralization is already under the critical `100%` threshold where pegged currency issuance value exceeds value of native token collateral. In such a case, the system will accept a liquidation deposit calculated as follows: `liquidator deposit` = `native token collateral` x `native token price on liquidation day` x (`100%` + `reward bid percentage`) + `outstanding fees`. Upon successful finalization, the liquidator receives the loan's entire backing in native token: `liquidation payment` = `native token collateral`. There will therefore be a non-zero leak of pegged currency due the bad loan.

Prolonged dispute state liquidation - If the system hasn't had `7 days` outside `Dispute` state within a maximum of `35 days` liquidation time limit since liquidation request, the system decides liquidation conditions based on the loan's original allocation. A `100%` allocation to one price feed will mean that liquidation decision will be solely made based on the price feed's latest price. A combination of say `3` price feeds with `33%`, `33%`, `34%` each will mean that liquidation condition will trigger if average of the feeds' latest prices warrants liquidation.

Liquidation reward/penalty - In order to incentivize timely liquidation, there needs to be a non-zero reward for liquidator to compete over. This reward is essentially a penalty that the owner of an undercollateralized loan, or the system as a whole would pay. As a result, we would also generally like competition for liquidations to push down the reward/penalty reducing the cost to an important stakeholder.

As a part of the liquidation request, the liquidator will send a bid for the reward they would be entitled to upon liquidation completion. The bid should be selected from the following values: `20%`, `15%`, `10%`, `5%`, `0%`. Lower bids supersede higher bids if made in the same system-defined day. Bids of equal value are then compared with respect to the request block time, where earlier bids supersede later ones. Equal bids of the same block time, are processed as described in the next section.

Front-running resistance - The system processes requests grouped and sorted by the timestamp under which the request has been submitted. The earliest timestamped group of requests are processed together for finalization, and if successful, are assigned a fraction of the liquidation process, in order of their deposit size, that is minimum of `100%` x `currency deposit` / `currency issuance` and `100%` / `number of remaining concurrent liquidation requests`

Price feeds

The main responsibility of price feed providers is the truthful **daily and delayed** reporting of **native token and pegged currency prices** to the system contracts. The report consists of 2 exchange rates, one for the native token in reference currency, and the other for the pegged currency also in reference currency. These rates are expected to be the market's global median prices based on volume. The system enforces the reporting of delayed historical prices for availability within 1 day (See maximum price feed delay), by imposing a penalties on any offending provider's revenue pool.

Delayed price reporting process

The price feed providers are responsible for reporting the designated median prices (by volume) daily, and have until the end of the next day to report. Every day, after gathering data and confirming the market activities of the previous day, and reporting the median prices for the following:

1. Daily median by volume rate of ETH in reference currency (ie US dollar $USD) - eg. $200.00 per ETH
2. Daily median by volume rate of pegged currency (ie Pegged US dollar) in reference currency - eg $1.01 per pegged US dollar

3. Day start time corresponding to blockchain time unit (time in seconds since unix epoch)

The time granularity of tracking feeds' historical prices is 1 day. This means all prices are reported as the median value for that whole day by volume, according to the system contract's definition of that day's start and end times.

For example, in our MVP implementation, this consists of ETH value in USD as well as pegged currency in USD as reported from exchange market activity. If there was `10,000` ETH traded for USD during the day, and `50%` of volume was under `$201.2` USD and the other `50%` was over `$201.2`, then `$201.2` should be reported as the ETH/USD price. Similarly, let's assume the Pegged USD was traded against ETH, with its median valued at `$0.9805` that day. The price provider would report the following and results:

- ETH/USD = `201.2`
- Pegged/USD = `0.9805`

The price feed provider should be aware of the system contract's timing for the start and end of each day an period, and plan accordingly to report the historical prices of each specific day, before the end of the subsequent day. For example the price for day 120 of the lifetime of the system contract, should be reported before the end of day 121.

Price feed values are results of formulas calculated from known market exchange prices and volumes. The ideal case is for all providers to have full access to the universe of all ***legitimate*** trades along with their volume and price, and good faith providers should strive to do so. However, due to many restraining factors, including uncertainty around authenticity of info from some sources or exchanges, as well as pure technical limitations, each provider will choose a specific set of samples they can depend on, at any given time. They should publish their methodology for compiling these historical prices, so the community can verify them in favor of higher level of trust and predictability for the ecosystem, as public good.

High trust providers are also expected to report something called **instant price** almost **immediately**, as they notice changes of over `1%`. This reporting by highly trusted providers, would be strongly encouraged by the ecosystem members (community) however there is no hard penalty imposed by the system itself.

Instant price reporting process

The price feed providers, that are in the trusted category, are responsible for reporting significant changes to the price of ETH in reference currency, within a minute of occurring. Significant changes are defined as any rise or drop of more than 5% since a previous report. Reporting changes of over 1% are recommended. The provider should monitor the latest median price by volume and upon a fall or rise of its value, send a on-chain request to their price contract with the value of:

1. Up to the minute median by volume rate of ETH in reference currency (ie US dollar $USD) - eg. $199.80 per ETH.
2. Price time corresponding to blockchain time unit (time in seconds since unix epoch)

Price feed revenue pool

Loan fees - Each price feed has a corresponding global revenue pool that increases in value as a result of an incoming portion of loan fees. See loan fees and *price feed revenue rate* under monetary variables for more details.

Price aggregation process - The total size of the price feed liquidity pool affects their weight during dispute resolution phase, as does the feed's allocation total

Dispute and instability penalties - During the price price aggregation process, any dispute or instability penalties issued to the price feed provider will come out of their liquidity pool. This is to incentivize providers into constructive behavior which is reporting the market prices truthfully.

Scheduled payouts - Price feed providers need to have a sustainable business model and so they are expected to have an income from the loan fees. This income takes the form of a scheduled set of payouts, that are allocated `2%` of the revenue pool every `1 week` and scheduled to be payed out in `25 weeks` assuming the underlying revenue pool contains that value. The payout is meant to be held in the system medium term, and is subject to slashing, in order to incentivize the price feed provider to maintain a constructive long term engagement with the platform. The payout is partial because there needs to be a significant liquidity pool to use in case of dispute or instability penalties.

Price feed delay and time resolution considerations

We have selected a specific price feed delay and time resolution of `1 day` as a result of considering multiple competing factors:

1. **Market conditions** - The level of demand based on competitive differences in the price feed market, determines the characteristics the system demands price feed. At one extreme, for example in the long term trading space, the market's demands may not exceed resolution and delay of `5 days`. At another extreme, such as for stock trading, the market may demand instantaneous (or in the order of `1 milliseconds`), in order to enable almost instantaneous transaction confirmation.
2. **Cost of reducing delay** - This is the cost of reducing price reporting delays on the blockchain, and is driven mainly by gas cost. At one extreme the provider may choose to record values `1x / day` with minimum gas cost (commonly `<1$`), or at the other extreme they may incur that gas cost on almost every block for instantaneous records.
3. **Cost of high resolution** - This is the cost of providing high resolution records, which is driven mainly by storage cost. At one extreme, the provider may choose to record prices every `5 days`, minimizing storage cost. At another extreme, the provider may choose to provide up to the block price records despite significant storage cost.

Savings account

Locking an amount of pegged currency in the savings contract, will allow for a collection of interest based on the variable daily savings rate (See ***savings interest rate*** under monetary variables), for the duration time of that locking. An call to withdraw the accumulated interest, calculates the to-date savings interest rate, savings interest amount and sets a timestamp for future interest withdrawal calls. An owner call to close the savings account, should be performed after withdrawing all applicable interest to that date.

Savings rate

Loan fees are also varied based on the global equilibrium price of the pegged currency, in order to incentivize increase or decrease the pegged currency supply based on changing the demand for pegged currency by consumers and thus affecting the equilibrium price. See ***savings interest rate*** under monetary variables.

Savings pool

Savings pool is a portion of pegged currency tokens held by the main system contracts, set to be payed out to money owners as interest, when they lock their tokens in a savings account contract for periods of time. A portion of the fee paid by loan takers is always transferred to this pool. Also, any penalties imposed on price feed are taken out of the offending provider's revenue pool and transferred into the savings pool. Payout is based on the ***savings interest rate*** (see savings rate and variable definition under monetary variables).

Notable constraint adjustments

In an ideal loan (debt position) system, one would expect perfect trust of the price feeds, very fine grained price feed resolution, and instantaneous response to loan taking requests or liquidation requests in appropriate conditions. However, given the constraints of highly decentralized protocols on public blockchain systems, as well as the urgent need for simplification, in order to reduce risks and increase efficiency, we choose to bend the ideal rules as long as these changes are communicated clearly to and are accepted by stakeholders, and as long as they result in a secure system overall.

Loan taking process adjustment

Normally the loan taking process should be expected to complete in one transaction, however during times of dispute between the rates provided by the system's price feed providers, it is reasonable to delay the loan taking process by a few days. Normal operation should be resumed on all other occasions.

Liquidation process adjustment

The most acceptable cases of liquidation from the perspective of a loan taker happen when there is a significant and non-intermittent drop in the value of the collateral, and they've had enough time to respond to it. We can adjust the current definition of acceptable liquidation to such cases, and ask the loan liquidators to take on the additional risk of having to predict if the drop is non-intermittent. We would however need to compensate the liquidator for the additional risk they are taking. It becomes therefore acceptable to condition liquidation upon the drop being persistent over the course of a few days and expect the loan liquidator to make a judgement on the likelihood of this persistence, and possibly take a profit.

Time constraint adjustments

One of the key insights that helps with efficiency and simplicity of this on-chain loan system is that using a collateral in smart contract to peg relatively stable currencies, does not require a high level of time sensitivity, in the following ways:

- **Liquidation process delay** - Given the intent and requirements from loan takers as well as loan liquidators, the liquidation triggering or the dispute process do not necessarily have to occur in real time, and can be delayed, even lasting for days, as long as they occur in a predictable manner.
- **Low time resolution** - One does not require a perfectly full resolution set of prices, in order to confirm a persistent drop in prices over a long period of time. The only case where higher resolution helps is determining the liquidation bid winner, as the first actor that submits a liquidation request right before passing solvency threshold.

Price feed penalties

All percent-based (%) penalties are enforced with respect to the corresponding provider's price feed revenue pool. The penalized value os transferred to the savings pool.

- **Missing price report penalty** - Missing any set of price values for each 1 day results in a penalty of 5% for that specific violation.
- **Reference price inaccuracy penalty** - Any one day that the historical reference currency price is reported to be 5% more or less than the system-wide median price, a penalty of 5% is imposed on the provider.
- **Pegged price inaccuracy penalty** - Any one day that the historical calculated peg currency price is reported to be 1% more or less than the system-wide median price, a penalty of 5% is imposed on the provider.
- **Savings account vote penalty** - Every week, if a consistent set of savings account owners vote to penalize a specific price feed address, the system will penalize that provider by a percent calculated by formula: `100% * total value of all voting savings accounts / total value of all savings accounts`

Appendix

Volatility

Historically, stable currencies have not shown a high level of short term volatility, where for example the price goes down 20% one day and goes back up a few minutes or hours. As such transaction delays have a much smaller chance of triggering costs to a stakeholder. There is often however clear longer term trends that can be observed with stable currency, at different points in time.

However one always needs to be prepared for short term volatility of the native value token due to unknown knowns such as general boom and bust cycles (or events), as well as unknown unknowns we can't even imagine. Although not likely to occur in case of a mature blockchain, it is entirely reasonable to consider an hour long drop in the order of 50%, or a sudden rise in the order of 400%, as a possibilities, and prepare for them.

Note on volatility of Ether

Since Ether (ETH) is the other side of the first viable product based on our proposed financial instrument, its volatility also affects our considerations. ETH in the recent years, due to its increasing uses as gas, speculative investment, and collateral, has been relatively more stable as compared to years before. Also due to its future staking use in Ethereum 2.0, we expect its long term volatility to decrease in general. However, it remains primarily a speculative asset subject to significant volatility.

Versioning - in case of (emergency) hard-forks

The ultimate goal of such a decentralized smart contract system is to be ownerless and live forever. That is, if the current open source implementation were the solution to our original stablecoin problem, there should be no subsequent version needed.

However, due to possibilities of future upgrades to the underlying blockchain itself, as well as due to the remote possibility that the system may, at some point, operate in some unexpected ways, we are required to at least consider the possibility of winding down this version in favor of a next one.

We are currently witnessing this type of transition with the SAI to DAI migration by Maker. There are a few lessons to be learned from this experience, in the remote case migration is needed.

Immutable price feed

One of the price feeds to the system on Ethereum can be constructed as a decentralized contract, based on market pricing information provided by the UniSwap V2 decentralized exchange. This ownerless, decentralized price feed would then automatically transfer any of its revenue payouts back into the loan system as part of the savings pool.

Ecosystem and game theory

The stablecoin system lives on the public blockchain, however the players interacting with it directly or indirectly, all live in the real world and will follow strategies and courses of action, based on their incentives and the changing ecosystem. In order to ensure the success of the stablecoin system, and the money product that it represents, we need to fully understand the ecosystem subsuming it and mitigate any foreseeable issues that may threaten the product's success.

Strategic dynamic changes

A few possibilities around shifting of strategic power between groups of actors:

- Conflict between price feed providers - Sub-group of price feed providers going rogue to eliminate another group from ecosystem
- Price feed providers collusion - Feed providers colluding to abuse the system through liquidations
- Price feed providers and loan taker collusion - Feed providers and loan takers colluding to take advantage of the money holders and users
- Influence consolidation through loan taking aggregators - Consolidation and influence of loan takers preferences through aggregators (like InstaDapp), and collusion with price feed providers for kickbacks
- Shortage of pegged currency - This can occur during wind-down of protocol in case of hard-fork, causing loss of collateral for loan takers

ToDo - Detail above dynamic changes along with mitigation steps considered by the system

Protocol changes

A few possible outcomes around changes to the protocol itself:

- Perpetual operation (no-fork) [default] - The same community will rely on the same protocol perpetually.
- Soft-fork - We end up with 2 sub-communities and 2 protocols adopted by each.
- Hard-fork - We end up with the same community moving to a new protocol. The old protocol will be transitioned and retired.

Perpetual operation (no-fork)

Summary: The same community will rely on the same protocol perpetually.

A successful general-purpose consumer money product is assumed to be held and used by large numbers of everyday people, which means the cost of altering the money during usage is extremely high to the individual people using it, as well as any other stakeholders. As a result, perpetual operation, meaning no disruptions to routine operation forever, is the ideal end-state of the system as designed, one that incurs the least costs in a large number of people having to switch from one protocol to another.

When the viability of the ecosystem are not in significant danger, this should be considered the default option. Of course, this is a subjective criteria, and is subject to the community's interpretation.

Soft-fork

Summary: We end up with 2 sub-communities and 2 protocols adopted by each.

There is always a non-zero probability that beliefs about operation of the ecosystem, amongst its actors could increasingly diverge in time, in which case, a community split is possible. In such a case, those whose beliefs are more aligned with the existing operations and the original parameters of the system are likely to stay with it, and incur less costs due to not having to switch to a new version. The existing sub-community however will incur a lot more cost.

The cost incurred by the sub-community leaving the ecosystem, includes but is not limited to, the cost of money users exchanging out of the pegged currency, the loan takers' cost of having to purchase sufficient pegged currency to cover loans (debt positions) and close them, as well as the cost of exiting price feed providers being slashed for the drops in their allocation numbers.

The original community will also experience a period of less certainty due to the large volume of pegged currency and deposits leaving the system, and incur other costs depending on size of the existing group.

Hard-fork

Summary: We end up with the same community moving to a new protocol. The old protocol will be transitioned and retired.

There is always a non-zero probability that the community as a whole may come to a conclusion that changes to the on-chain protocol are required in order for the ecosystem to remain viable. In such a case, the community as a whole would have to coordinate simultaneous transition to a new version of the system, one that uses parameters agreed upon by the community as a whole.

Given that the system is immutable and holds a large amount of state, in the form of value, transitioning out of it will be very costly, including but not limited to:

- **Uncertainty and unpredictability** is a significant cost to an ecosystem that operates optimally based on predictability, trust, and truthful price reporting as a schelling point for all stakeholders. Any effective mitigation to the this will include:
    - An effective transition plan, one that considers mitigations to possible outcomes, and ensures the interests of all stakeholders are preserved as much as possible.
    - Clear communication of the transition to all stakeholders, instructions on how to proceed.
- **Price volatility** is inevitable given that transition out of the current system requires movement of large amounts of value out of the system, in form of exchanging out of the pegged currency money, and in form of closing loans (debt positions) using large amounts of purchased pegged currency. Higher amounts of uncertainty, as mentioned above, will adversely affect volatility.
- **Money users' overhead** - very high cost of having to even care about this, cost of effort needed to exchange out of the pegged currency.
- **Money users' exchange costs** - Exchanging tokens has a non-zero fee regardless of the exchange used, this is no different.
- **Loan takers' overhead** - high cost of having to care about this in the first place, as well as the effort to purchase sufficient pegged currency to cover loans (debt positions) and close them.
- **Loan takers' exchange costs** -

- **Price feed provider slashing** - the cost of exiting price feed providers being slashed for the drops in their allocation numbers, as well as losses of time based payouts.
- **Possibility of un-closable loans due to lost pegged currency** -
- ToDo - others ...

Community

As a decentralized system and ecosystem, the healthy operation of {{PegLoan}} will depend on a community of users, loan takers, feed providers and others. The community's potential points of common belief are:

- **On-chain governance minimization** - Skepticism of governance through tokens, and belief in minimizing it.
- **Market truth as schelling point** - The common belief that truth is the ecosystem's schelling point. Community members and ecosystem participants believing they can sustainably benefit from helping operate and interacting with the system, and that truthfully reported price feed information is in everyone's interest.
- **Crypto-sovereignty** - The belief that sufficiently decentralized public blockchains as a whole from an unconventional sovereign jurisdiction on the internet, one that lacks many of the restrictions and friction points of today's geographical jurisdiction, and is more suitable for free economic activity. This consists of a few other shared beliefs around censorship resistance, decentralization and trust minimization.
- **Crypto-nationalism** - (a better and alternative definition of token maximalism) is the sometime exclusive allegiance to a specific public blockchain. It consists of the following set of beliefs:
    - Belief and feeling of belonging to that specific public blockchains community
    - The specific blockchain represents the best potential unconventional sovereign jurisdiction on the internet, which is akin to a nation.
    - More extreme versions of crypto-nationalism will advocate for exclusive citizenship in many contexts, at the cost of other public blockchains.
    - The native digital asset of this specific public blockchain is the most trust-minimized and is a superior store of value.
    - The specific blockchain will become the most ubiquitously used public blockchain in the future.