Lava Loans Protocol

Lava Global Inc

May 12, 2023

Abstract

Serving a role as a digital store of value, bitcoin is a strong candidate for loan collateral. However, there exist no good options for users that wish to use native-bitcoin as collateral for loans without trusting a custodian. We present a solution that adapts methods from discreet log contracts to enable people to borrow assets against native-bitcoin as collateral. One of the most important use-cases of the Lava Loans Protocol is that it enables people to borrow stablecoins against native-bitcoin in the most trustless and transparent way currently possible.

1 Background

People want to borrow against their bitcoin for many reasons: to maximize exposure to bitcoin, to minimize tax liability, and to arbitrage the difference between the borrow rate and bitcoin appreciation rate. Since borrowing is almost always over-secured, people have to give up custody of their collateral to centralized parties when borrowing, and thus borrowers end up having no guarantees on how their collateral is stored or if it is even being stored. As a result, we see billions of dollars in user funds have been lost by custodians who took excess risk. It has been shown that custodians can and will most likely lose user funds. People need more transparent and secure ways to interact with financial primitives for bitcoin and crypto.

Whereas people still want to borrow, they now want solutions that are more trustless and transparent. Specifically, borrowers have the following objectives:

1. **Trust-minimized, atomic initiation of the loan.** There should be no point where the lender holds the collateral and can choose not to give the user the loan amount.
2. **Trust-minimized loan repayment.** Borrowers should be confident that if they repay their loan, their loan repayment will be accepted and that they will receive their collateral back.
3. **Trust-minimized loan liquidation.** All over-secured loans are callable, meaning the collateral can be liquidated and sent to the lender. Borrowers want confidence that they will not be liquidated unfairly, meaning that they will only be liquidated if the collateral value actually goes below a certain loan-to-value threshold determined at loan initiation.
4. Under the case of loan expiration, where the loan term closes, borrowers want confidence that they will receive their collateral back minus what is owed to the lender.
5. Borrowers desire low interest rates and fast, easy user experiences.

Lenders are also a key part of any borrowing system, and until now, they have not been able to lend against native-bitcoin collateral without either taking custody of user funds or lending against wrapped bitcoin. Lenders have the following objectives:

1. Lenders want to make sure in an over-secured loan system that the collateral is instantly callable. This is to ensure that the lender is taking the minimal risk, essentially only underwriting technology risk and liquidity risk.
2. Lenders want to ensure they can earn a high enough yield for their risk assessment, and they also want access to liquidity to remove funds if necessary.

Out of any borrowing solution that currently exists, the Lava Loans Protocol best meets the objectives of the borrowers and lenders while creating a radically faster, more trustless, more transparent, and more secure user experience.

2 Introduction

Let's start by defining some terms:

1. "Loan capital asset" is synonymous with a borrowed asset.
2. r is the compounded-daily interest rate at which the loan is issued.
3. T is the number of days after loan initialization that the loan will expire.
4. t is the number of days that have transpired since loan initialization.
5. V is the amount of bitcoin collateral.
6. L is the borrowed amount denominated in the loan capital asset.

Let $X(t)$ be the exchange rate between bitcoin and the loan capital asset at time t–this value is provided by a set of oracles. The loan-to-value ratio is defined as the following:

$$\text{LTV}_{V,X,L}(t) = \frac{V \cdot X(t)}{L \cdot \left(\frac{r}{365}\right)^{365}}$$

Recognize that this accrues interest with respect to the loan capital asset. The "staked amount" is computed as the following:

$$S = L \cdot \left(1 + \frac{r}{365}\right)^{T} - L$$

This is the total amount of interest the lender could earn over the course of the loan. The purpose of this staked amount will be clearer in section 3.2.3.

3 Model

1. Borrower and lender are matched.
2. Loan is initialized with bitcoin collateral locked on layer 1 and borrowed asset redeemed by the borrower.
3. Contract Execution:
   (a) If the loan-to-value ratio rises too high (i.e. $\text{LTV}(t) > 1-\epsilon$ for small ϵ), then the collateral is immediately redeemable by the lender. This requires the oracles to attest to a liquidation-inducing price of bitcoin.
   (b) User pays the borrowed asset back, including interest before the loan period ends to a smart contract, revealing a secret that will enable the unlocking of collateral.
   (c) If the user fails to pay back the loan before the specified end date, the loaned amount with accrued interest will be redeemable by the lender and the excess collateral will be redeemable by the borrower. This requires the oracles to attest to a price of bitcoin that determine how much is owed to both parties.
   (d) If all else fails (oracles do not attest to the price of bitcoin).

3.1 Initializing the contract

The process of contract initialization is extremely akin to that of atomic swaps. The borrower and lender come with bitcoin collateral and loan capital, respectively. Whereas the custody of the assets is indeed swapped with atomic swaps, in the context of lending, the collateral must instead move to a specified multi-signature address between the lender and the borrower.

3.2 Contract Execution

Contract execution transactions (CETs) spend from this multisig and fall into one of four cases: liquidation, repayment, expiration and refund. These CETs are constructed in the same way as in discreet log contracts. To learn more about discreet log contracts, check out the specification here. Out of the specification, it is important to understand ECDSA-Adaptor, NumericOutcomeCompression, MultiOracle and Transactions. The rest of the paper assumes knowledge of these.

3.2.1 Liquidation

Fix a time t and reasonably small ϵ. If liquidation occurs when

$$\begin{aligned} LR(t) &< 1+\epsilon \\ \frac{C \cdot X(t)}{L \cdot \left(1+\frac{r}{365}\right)^t} &< 1+\epsilon \\ \implies X(t) &< \frac{(1+\epsilon) \cdot L \cdot \left(1+\frac{r}{365}\right)^t}{C} \end{aligned}$$

implies that $X'(t) = \frac{(1+\epsilon) \cdot L \cdot \left(1+\frac{r}{365}\right)^t}{C}$ is the maximum asset-denominated exchange rate at which liquidation occurs on the day t.

For each day, there is a liquidation CET that yields all of the collateral to the lender. The borrower's signature is encrypted behind an adaptor signature. If the set of oracles attests to a price of bitcoin below $X'(t)$, then the lender will be able to decrypt the adaptor signature into the borrower's signature and broadcast the liquidation CET.

The lender now needs to reclaim his staked amount S (the purpose of which will be explained in the next section). The lender takes the signature for the liquidation, `XOR`s it with a specified "perturbed preimage" for the particular day and uses that value as the valid preimage for unlocking his stake. The process of verifying the ability to do this is done in section 3.1.

3.2.2 Loan Repayment

To initiate repayment, the borrower sends the principal plus interest accrued to the alt-chain smart contract. This alt-chain smart contract virtual machine must be expressive enough to assert that the correct amount has been repaid, given the time elapsed since the start of the loan.

Let's assume the lender cooperates. He "accepts" repayment of the loan by revealing a preimage to a SHA256 hash, which the borrower can use as the lender's signature for the DLC multisig to reclaim his collateral on bitcoin.

It is important that the lender cooperates when the borrower attempts to repay the loan—if he chooses not to, he could allow the loan to expire and accrue extra interest that the borrower does not need to pay. Thus, the lender must put up some stake to compel himself to work with the borrower. This stake should be equal to the amount of interest the borrower would pay back if he held the loan out to expiry, that is S as defined in the introduction. When the lender reveals the preimage, he will receive both the staked amount S and the principal + interest that the lender paid back.

3.2.3 Loan expiration and termination

At loan expiration, because no loan repayment has occurred, bitcoin is due to both the borrower and lender. Specifically $Q := L \cdot \left(1+\frac{r}{365}^T\right) / X(T)$ and $V - Q$ is due to the lender and borrower respectively.

This is paid out to the lender and borrower exactly as a traditional DLC would. An important parameter we make native to the loans protocol is the "attested price bucket size". This is the size of buckets of bitcoin price values that share a single CET. If this value is 5000, then a single CET would be constructed for the attested price of bitcoin in the range 0-5000, 5001-10000, 10001-15000, etc. Typically this is set to a much smaller value. By increasing this value, we decrease the granularity of the contract execution, implying that fewer adaptor signatures need to be created, verified and stored. Intuitively, the time and space complexity of adaptor signature creation/verification and storage is

inversely related to the "attested price bucket size". That is, if we double the value of of "attested price bucket size", we should halve our time and storage usage.

Furthermore, if the crosses a timelock threshold (i.e. expires) on the alt-chain, then the lender should be able to redeem his stake amount S on the alt-chain smart contract.

3.2.4 Refund

If all else fails and none of the three execution events occur, then the borrower is entitled to redeem all of his collateral after a locktime (e.g. 2 after normal expiration).

The lender will likewise be able to claim his staked amount S for the same reason mentioned in the previous subsection on loan expiration.

4 Oracle Considerations

There are several parameters regarding the oracles that must be selected ahead of time:

- **The set of oracles:** the specific oracles in charge of the attestation of the price of bitcoin.
- **Oracle threshold:** the number of oracles that must agree for a liquidation or expiration contract execution event to occur. For instance, if the set of oracles has size 5 and the oracle threshold is 3, then any three of the five oracles must agree.
- **Oracle error tolerance:** the extent that oracles must generally agree. This parameter is actually two numbers, `min_support` and `max_error` where `min_support` is the minimum difference in price attestations that must be tolerated while `max_error` is the max difference that can be tolerated. Both numbers must be powers of two and `min_support` $<$ `max_error`. The price differences in the range of the two numbers are not guaranteed to be supported.
- **Attested price bucket size:** discussed in section 3.2.3.

5 Risks

There are three notable risks worth discussion:

1. **Oracle risk.** The set of oracles could collude with either of the parties. If the oracles collude with the lender, they could cause a liquidation event to execute even if the collateral is worth well over the loan value, causing the borrower to lose the amount: collateral value - loan value. If the oracles collude with the borrower, they could prevent liquidations from occurring, attest to a high price of bitcoin at expiration such that the majority of the collateral is unduly returned to the borrower. They could also simply not attest at all and force a refund CET.
2. **Loan capital risk.** Let us say the loaned capital is a stablecoin that we realize is not backed 1:1 with dollars, and it loses its peg. Paying back the stablecoin amount becomes much easier for the borrower, and he is very inclined to do so. This is further exacerbated by the fact that that the oracles are attesting to the price of bitcoin denominated in dollars, not stablecoin (when loan expires the value of the loan in dollars is send the the lender). Consequently, the brunt of the stablecoin risk is put on the lender.
3. **Altchain risk.** If the altchain fails, then the borrower will have no way to repay his loan. This means the loan will necessarily expire, and some of the collateral will go to the lender. If the borrower had sold his stablecoin for a different asset off the altchain (say fiat US Dollars), then he is comfortable. However, because the altchain has failed, the lender will never be able to reclaim his staked amount S, which can be assumed to be lost.

While these risks exist, there are many ways to respond to them:

1. **Oracle risk.** Oracles for the Lava Loans Protocol are not natively aware that a contract is relying on their date. They are blind to contracts. Oracles also have little incentive to misbehave. They can't receive borrower/lender funds as funds only move to borrowers or lenders. Oracles are also negotiated at loan initiation so borrowers and lenders are both aware of the oracles involved in a loan contract. We can also increase the set of oracles to make the contracts as trust-minimized as possible by adding redundancy. To make this easier, Lava has released an open-source oracle implementation called sibyls.

2. **Loan capital risk.** Lenders and borrowers can decide which stablecoins they want to interact with, so they can assess risks prior to the initiation of their contracts. If a loan interfaces with USDC, USDT, or another 1:1 backed stablecoin, funds can also be recoverable via contacting Circle, Tether, or the respective custodian.

3. **Altchain risk.** Lenders and borrowers can decide what chain/network they would like the stablecoins to live on before loan initiation. Some funds can also be recovered by contacting the fiat custodian.

6 Conclusion

We present this protocol to bring more trustless and transparent functionality to bitcoin. The technology we use to build the protocol can be adapted to build other functionality that uses native-bitcoin. Bitcoin collateral can also live on lightning, not just layer 1. Lava believes the future of sovereign finance will involve sovereign functionality being built for sovereign money.