Wrapless: Bitcoin lending without wrapping services

Rarimo Protocol

Oleksandr Kurbatov, Yaroslav Panasenko, Pavel Kravchenko
ok@distributedlab.com

Mar, 2025

Abstract

1 Introduction

Now, there is only one action people can take with bitcoins in a trustless manner: transfer. Bitcoin programmability is limited and doesn't allow us to launch popular DeFi protocols operating with native bitcoins. So, the popular approach is wrapping bitcoins and using them in the existing DeFi infrastructure. The problem is the existing wrapping mechanisms are either centralized or federated with m-of-n trust assumptions. Despite the massive potential of BitVM2 technology [Lin24], its potential depends on the acceptance of covenants.

We tried to design an approach that allows collateralization of bitcoins without wrapping. This paper describes the protocol that allows locking BTC in the "credit channel" and providing the credit on the other blockchain platform in the way the credit's payoff (which can be partial) leads to unlocking the appropriate amount of BTC.

1.1 Notation

$\mathbb{G}$ a cyclic group of prime order p written additively, $G \in \mathbb{G}$ is the group generator. $a \in \mathbb{F}_p$ is a scalar value and $A \in \mathbb{G}$ is a group element.

We define a bitcoin transaction as $\mathsf{TX}\{(\mathsf{id}, \mathsf{i}, \mathsf{proof})^{(n)}; (\mathsf{a}\text{₿}, \mathsf{cond})^{(m)}\}$ with n inputs and m outputs, where id is the reference to the previous transaction, i - output's index, proof - the list of data which is needed to transaction spending, a - the number of coins in the output, cond - scriptPubKey conditions. Additionally, we use $\sigma_{\mathsf{a}}(m)$ notation for the signature that is verified by the public key $\mathsf{P_a}$ according to the message m.

1.2 Model

We expect to have a lending/borrowing system that works the following way:

1. Alice owns a₿ and wants to take a loan of b USDC (issued on Ethereum, for instance) with BTC as collateral

2. Bob owns b USDC and is ready to provide it to Alice if she will return $(b + f)$ USDC divided into n loan installments during a particular time period (the deadline $T_i, i \in [1, n]$ for each installment is known in advance). Let's note that we leave the mechanism for organizing the credit market out of the paper.

3. Alice locks BTC in the way she receives the full body of credit in an atomic way.

4. If Alice pays back the particular loan part $(\frac{b+f}{n})_i$ before T_i and Bob accepts it:

    (a) Bob can take $\frac{b+f}{n}$ USDC from the contract

   (b) Alice can take $\frac{a}{n}$Ƀ from initially locked in the atomic way

5. If, at any point in time,e Alice doesn't return the part of the credit or Bob doesn't accept that:
   (a) Bob can take the BTC is left
   (b) Alice can take BTC that was covered by loan installments and not return the initial loan amount is left

2 Related work

2.1 SPV connector

The SVP connector [Kur24] is a smart contract that synchronizes the whole Bitcoin history to the blockchain when it's deployed. The idea of the connector is to verify all block headers (that any user can provide) and resolve reorganizations by following Bitcoin protocol rules. Using the synchronized SPV contract, the user can prove that a particular transaction was confirmed in the Bitcoin mainnet and trigger the defined action on the targeting blockchain. We will refer to $\pi_{\mathsf{SPV}}(\mathsf{tx}) \to \mathbb{B}$ as the SPV proof that the particular transaction tx was included in the Bitcoin blockchain.

2.2 Payment channels and Lightning Network

Payment channels [PD16] is a technology that allows counterparties to lock their funds and send instant payments within this channel without posting on-chain intermediate transactions. We will refer to the simplified payment channel construction to reuse it later for our "credit channels":

Algorithm 1 Lightning Network Channel Opening and Closing

Condition: Alice and Bob want to establish a bidirectional payment channel using a funding transaction. Alice provides aɃ to the channel, and Bob – bɃ. The channel can be closed cooperatively or unilaterally.

Flow:

```
1. Alice and Bob generate their respective keys:
       $sk_a \xleftarrow{R} \mathbb{F}_p, \quad P_a = sk_a G$
       $sk_b \xleftarrow{R} \mathbb{F}_p, \quad P_b = sk_b G$
2. Alice and Bob construct a funding transaction:
       $\mathsf{TX_f}\{(\mathsf{prev_A}, \mathsf{i_A}, \sigma_\mathsf{A}(\mathsf{TX_f})), (\mathsf{prev_B}, \mathsf{i_B}, \sigma_\mathsf{B}(\mathsf{TX_f})); ((\mathsf{a}+\mathsf{b})Ƀ, \mathsf{musig}(\mathsf{P_a}, \mathsf{P_b}))\}$
3. Alice and Bob create initial commitment transactions (before broadcasting $\mathsf{TX_f}$):
       $\mathsf{TX_{ca_1}}\{(\mathsf{TX_f}, 1, \sigma_\mathsf{B}(\mathsf{TX_{ca_1}})); (\mathsf{a}Ƀ, \mathsf{addr_a}), (\mathsf{b}Ƀ, \mathsf{HTLC} \vee \mathsf{addr_b})\}$
       $\mathsf{TX_{cb_1}}\{(\mathsf{TX_f}, 1, \sigma_\mathsf{A}(\mathsf{TX_{cb_1}})); (\mathsf{a}Ƀ, \mathsf{HTLC} \vee \mathsf{addr_a}), (\mathsf{b}Ƀ, \mathsf{addr_b})\}$
4. Alice and Bob exchange signatures for their respective commitment transactions
5. Alice and Bob broadcast $\mathsf{TX_f}$ to open the channel
6. For each update, new commitment transactions $\mathsf{TX_{ca_i}}$ and $\mathsf{TX_{cb_i}}$ are generated, signed, and exchanged
   before revoking old ones.
7. Closing the channel.
```

The channel can be closed in one of the following ways:

- **Cooperative Close:** Alice and Bob agree on a final state and create $\mathsf{TX_{cc}}$, which is broadcast to settle balances

$$\mathsf{TX_{cc}}\{(\mathsf{TX_f}, 1, \sigma_\mathsf{AB}(\mathsf{TX_{cc}})); (\mathsf{a'}Ƀ, \mathsf{addr_a}), (\mathsf{b'}Ƀ, \mathsf{addr_b})\}$$

- **Unilateral Close:** If one party stops responding, the other broadcasts the latest valid commitment transaction

- **Revoked Transaction Penalty:** If a revoked commitment transaction is broadcast, the counterparty can claim all funds using a pre-signed penalty transaction

3 Wrapless protocol

Combining the principles of the Lightning network and SPV contract, we can build a flow that presumes to provide the loan collateralized by locked on-chain BTC without wrapping and supplying it to the lending protocol deployed on any VM network.

Algorithm 2 Wrapless protocol

Condition: Alice wants to lock 1₿ as collateral and receive the 100,000 USDC loan from Bob on Ethereum. Bob wants to receive an additional 10,000 USDC as a fee for the loan facility. So, Alice needs to do 10 installments, 11,000 USDC each, to cover the loan and the fee.

Flow:

1. Alice and Bob construct the funding transaction for the payment channel as:

$$\mathsf{TX_f}\{(\mathsf{prev_A}, \mathsf{i_A}, \sigma_\mathsf{A}(\mathsf{TX_f})), (\mathsf{prev_B}, \mathsf{i_B}, \sigma_\mathsf{B}(\mathsf{TX_f})); (1₿, \mathsf{musig}(\mathsf{P_a}, \mathsf{P_b}))\}$$

2. Alice and Bob create 11 commit transactions ($i \in [0, 10]$) that represent all states before and after each loan installment:

$$\mathsf{TX_{ca_i}}\{(\mathsf{TX_f}, 1, \sigma_\mathsf{B}(\mathsf{TX_{ca_i}})); ((0.1\mathsf{i})₿, \mathsf{addr_a}), ((1 - 0.1\mathsf{i})₿, \mathsf{HTLC} \vee \mathsf{addr_b})\}$$
$$\mathsf{TX_{cb_i}}\{(\mathsf{TX_f}, 1, \sigma_\mathsf{A}(\mathsf{TX_{cb_i}})); ((0.1\mathsf{i})₿, \mathsf{HTLC} \vee \mathsf{addr_a}), ((1 - 0.1\mathsf{i})₿, \mathsf{addr_b})\}$$

3. Alice and Bob exchange signatures for $\mathsf{TX_{ca_0}}$ and $\mathsf{TX_{cb_0}}$
4. Bob deploys the escrow contract on Ethereum and deposits 100,000 USDC to it.

   The deposit can be returned after T_0 (absolute timelock) if Alice doesn't provide $\pi_\mathsf{SPV}(\mathsf{TX_f}) \rightarrow 1$

   If $\pi_\mathsf{SPV}(\mathsf{TX_f}) \rightarrow 1$ was provided before T_1, the deposit can be claimed and withdrawn by Alice

   The escrow contract is set with all commit transactions

   There are three function related to settlement: **payoff()**, **claim()** and **revoke()**

   There are $T_i, i \in [1, 10]$ - absolute time values that indicate deadlines for each installment
5. Alice makes a ith loan installment by calling **payoff()** function, sending 11,000 USDC, and revealing the signature for $\mathsf{TX_{ca_i}}$.
6. If Bob accepts the installment, he calls **claim()** function, sends the signature for $\mathsf{TX_{cb_i}}$ and secret committed in the $\mathsf{TX_{cb_{i-1}}}$.
7. Alice calls **revoke()** function and provides the secret committed in $\mathsf{TX_{ca_{i-1}}}$

If all steps were formed as described, Alice and Bob settled on a new commit transaction with updated amounts on outputs.

---

4 Security claims

- If Alice doesn't return the loan, Bob can take BTC is left in the payment channel
- Alice and Bob can close the credit channel at any point in time, taking BTC according to the last agreed commitment transactions
- There is a risk of Bitcoin price changes that can lead to a state where it's not profitable for Alice/Bob to pay/accept the loan
- If the external blockchain is destroyed, Alice can still return all BTC left in the channel to her address

5 Ideas

1. The locked BTC can provide liquidity to the Lightning network and be an additional source of income based on fees in payments conducted through their channel.
2. Escrow contracts can be integrated with different identity applications, allowing one to calculate the credit cost individually.
3. An escrow contract can be integrated into the Uniswap wBTC/USDC pool, taking the actual ratio and recalculating the credit based on the BTC price within the previous period.
4. Basically, the proposed model is collateralized debt positions (CDP) with additional questions to be managed like possible interest rate model, liquidation process in case the loan goes overdue, and managing collateralization ratio for our BTC collateralized loan.

6 Acknowledgments

We want to thank `0x09c0034602c8a4ce6174f6c6c5f56d9a32581592`, who decided to be anonymous, but still, we couldn't help but note his impact on work on this idea.

References

[PD16] Joseph Poon and Thaddeus Dryja. "The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments". In: (2016). URL: https://lightning.network/lightning-network-paper.pdf.

[Kur24] O. Kurbatov. "EXTENDING THE SPV CONTRACT CONCEPT WITH PRIVACY GADGETS". In: *Telecommunication and Information Technologies* 83 (July 2024). DOI: 10.31673/2412-4338.2024.024961.

[Lin24] Robin Linus. "BitVM2: Extending Bitcoin's Computation Model". In: (2024). URL: https://bitvm.org/bitvm2.pdf.