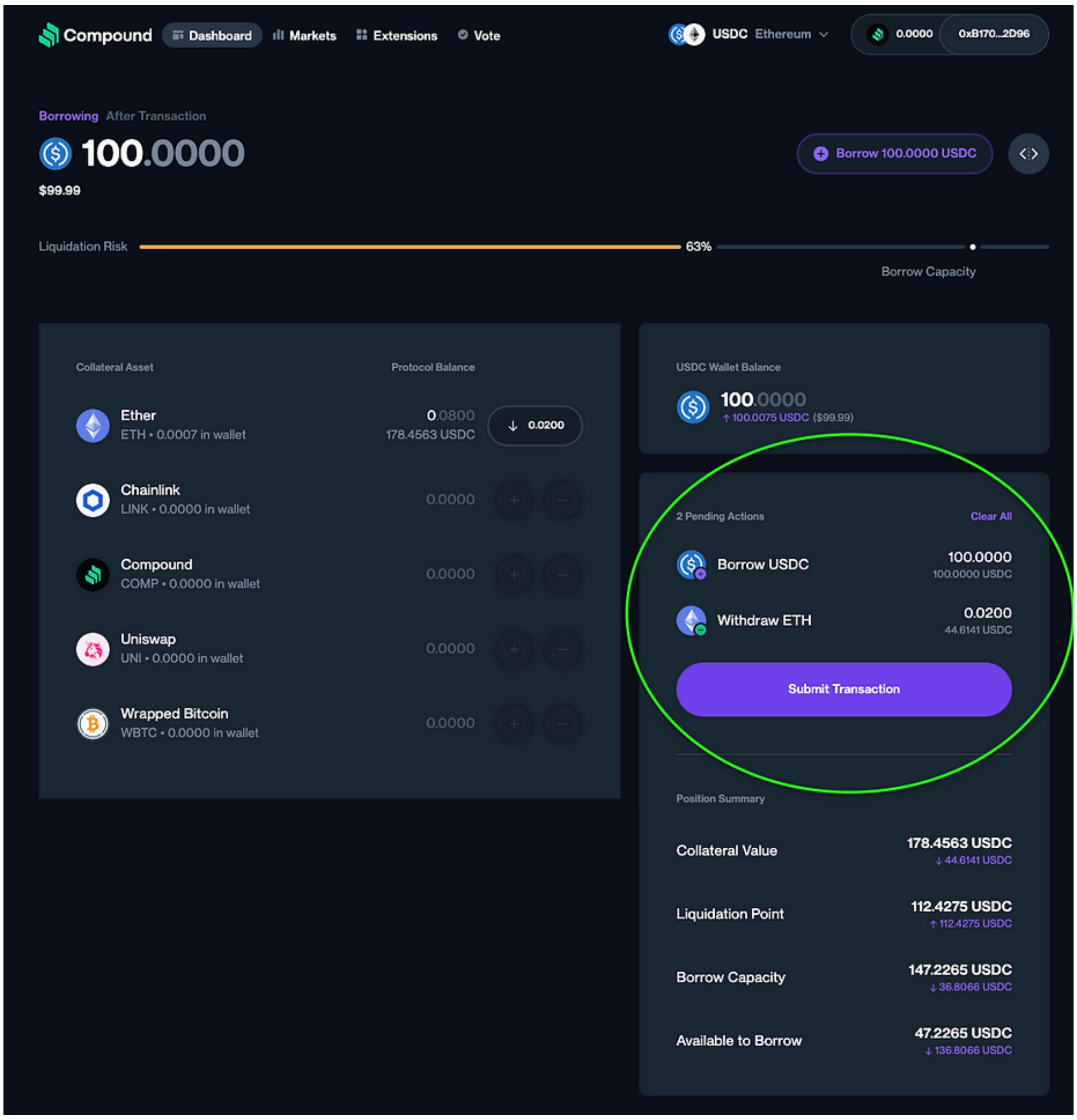
Jeffrey Scholz Jan 9 3 min read

Bulkers in Compound V3

The bulker contracts in Compound V3 are multicall-like contracts for batching several transactions.

For example, if we wanted to supply Ether, LINK, and wBTC as collateral and borrow USDC against it in one transaction, we can do that.

We can also reduce the collateral holdings and withdraw a loan in one transaction as the screenshot below demonstrates. This of course assumes that we stay within the collateral factor limits.



100.0000
$99.99

Collateral Asset	Protocol Balance
Ether ETH • 0.0007 in wallet	0.0800 178.4563 USDC (↓ 0.0200)
Chainlink LINK • 0.0000 in wallet	0.0000
Compound COMP • 0.0000 in wallet	0.0000
Uniswap UNI • 0.0000 in wallet	0.0000
Wrapped Bitcoin WBTC • 0.0000 in wallet	0.0000

USDC Wallet Balance
100.0000
↑ 100.0075 USDC ($99.99)

2 Pending Actions Clear All

Borrow USDC	100.0000 100.0000 USDC
Withdraw ETH	0.0200 44.6141 USDC

Position Summary

Collateral Value	178.4563 USDC ↓ 44.6141 USDC
Liquidation Point	112.4275 USDC ↑ 112.4275 USDC
Borrow Capacity	147.2265 USDC ↓ 36.8066 USDC
Available to Borrow	47.2265 USDC ↓ 136.8066 USDC

invoke()

- supply an ERC 20 (ACTION_SUPPLY_ASSET)
- supply ETH (ACTION_SUPPLY_NATIVE_TOKEN)
- transfer an asset (ACTION_TRANSFER_ASSET), see our article on how Compound V3 behaves like a rebasing ERC 20 token to see how this works
- withdraw an ERC 20 (ACTION_WITHDRAW_ASSET)
- withdraw ETH (ACTION_WITHDRAW_NATIVE_TOKEN)
- claim accumulated COMP rewards. The underlying function claimReward will interact with the reward contract instead of the main lending contract (Comet.sol).

Invoke will loop through the actions and call Comet (or the rewards contract) with the arguments supplied.

Although it could have been more gas efficient to put this code into the main contract, using msg.value in a loop, especially when invoking delegatecall, is not safe. See the practice problem at the end of this article.

Compound is careful to make sure msg.value is deducted rather than re-used, which could lead to double spending — see the yellow boxes.

This arguably could have been gas optimized by having the actions decided with a 1 byte indicator rather than a 32 byte ascii string indicator.

The bulkers are non-custodial

Rescuing tokens

In the case where users accidentally transfer tokens to the bulker, an admin can remove the tokens. Note we have separate functions for remove ETH that is stuck and ERC 20 tokens that are stuck.

Handling non-standard ERC 20 tokens

One of the most common deviations from the ERC 20 token specification is not returning a boolean value. Several ERC 20 tokens do not return a boolean but revert in the failure case.

To handle both situations when dealing with ERC 20 assets, Compound implements the code below:

Essentially, we succeed on the cases where the (token does not revert AND returns nothing) OR (token does not revert AND returns true). We fail on cases where (token reverts) OR (token does not revert AND returns false).

The IERC20Nonstandard interface simply defines an ERC20 token that does not return any booleans.

Practice problems

Bad things can happen when using msg.value inside a loop. See these two practice problems:

1. RareSkills Riddles MultiDelegateCall
2. DamnVulnerableDefi Free Rider

Learn More with RareSkills

Please see our web3 bootcamp to learn more.

Recent Posts See All