LIDO FINANCE ANCHOR COLLATERAL SMART CONTRACT AUDIT

July 30, 2021

MixBytes()

CONTENTS

1. INTRODUCTION

1.1 DISCLAIMER

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of LIDO Finance. If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

1.2 PROJECT OVERVIEW

Anchor Vault is a smart contract that allows to convert rebasing stETH token into a constant-balance bETH token and periodically send all accrued stETH rewards to the Terra blockchain through a bridge. The bETH token will be used as a collateral in the Terra Anchor protocol.

1.3 SECURITY ASSESSMENT METHODOLOGY

A group of auditors are involved in the work on the audit who check the provided source code independently of each other in accordance with the methodology described below:

01 Project architecture review:
> Reviewing project documentation
> General code review
> Reverse research and study of the architecture of the code based on the source code only
> Mockup prototyping

Stage goal:
Building an independent view of the project's architecture and identifying logical flaws in the code.

02 Checking the code against the checklist of known vulnerabilities:
> Manual code check for vulnerabilities from the company's internal checklist
> The company's checklist is constantly updated based on the analysis of hacks, research and audit of the clients' code
> Checking with static analyzers (i.e Slither, Mythril, etc.)

Stage goal:
Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flashloan attacks, etc.)

03 Checking the code for compliance with the desired security model:
> Detailed study of the project documentation
> Examining contracts tests
> Examining comments in code
> Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit
> Exploits PoC development using Brownie

Stage goal:
Detection of inconsistencies with the desired model

04 Consolidation of interim auditor reports into a general one:
> Cross-check: each auditor reviews the reports of the others
> Discussion of the found issues by the auditors
> Formation of a general (merged) report

Stage goal:
Re-check all the problems for relevance and correctness of the threat level and provide the client with an interim report.

05 Bug fixing & re-check:
> Client fixes or comments on every issue
> Upon completion of the bug fixing, the auditors double-check each fix and set the statuses with a link to the fix

Stage goal:
Preparation of the final code version with all the fixes

06 Preparation of the final audit report and delivery to the customer.

Findings discovered during the audit are classified as follows:

FINDINGS SEVERITY BREAKDOWN

Level	Description	Required action
Critical	Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party	Immediate action to fix issue
Major	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.	Implement fix as soon as possible
Warning	Bugs that can break the intended contract logic or expose it to DoS attacks	Take into consideration and implement fix in certain period
Comment	Other issues and recommendations reported to/acknowledged by the team	Take into consideration

Based on the feedback received from the Customer's team regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The project team is aware of this finding. Recommendations for this finding are planned to be resolved in the future. This finding does not affect the overall safety of the project.
No issue	Finding does not affect the overall safety of the project and does not violate the logic of its work.

1.4 EXECUTIVE SUMMARY

The audited scope implements the RewardsLiquidator, InsuranceConnector and BridgeConnectorShuttle contracts installed as delegates to the AnchorVault contract. These contracts can be replaced by the vault admin. The BridgeConnectorShuttle contract is designed to interoperate with the Shuttle Ethereum-Terra bridge.

1.5 PROJECT DASHBOARD

Client	LIDO Finance
Audit name	Anchor Collateral
Initial version	8384738b043ea139bb0d01f0ccd446204fbfd43e
Final version	c24beb01afd7f6397e78062dd9a46906d38bc08b
Date	July 06, 2021 - July 30, 2021
Auditors engaged	2 auditors

FILES LISTING

AnchorVault.vy	https://github.com/lidofinance/anchor-collateral-steth/blob/8384738b043ea139bb0d01f0ccd446204fbfd43e/contracts/AnchorVault.vy
BridgeConnectorShuttle.vy	https://github.com/lidofinance/anchor-collateral-steth/blob/8384738b043ea139bb0d01f0ccd446204fbfd43e/contracts/BridgeConnectorShuttle.vy
InsuranceConnector.vy	https://github.com/lidofinance/anchor-collateral-steth/blob/8384738b043ea139bb0d01f0ccd446204fbfd43e/contracts/InsuranceConnector.vy
RewardsLiquidator.vy	https://github.com/lidofinance/anchor-collateral-steth/blob/8384738b043ea139bb0d01f0ccd446204fbfd43e/contracts/RewardsLiquidator.vy
bEth.vy	https://github.com/lidofinance/anchor-collateral-steth/blob/8384738b043ea139bb0d01f0ccd446204fbfd43e/contracts/bEth.vy

FINDINGS SUMMARY

Level	Amount
Critical	0
Major	0
Warning	1
Comment	4

CONCLUSION

Smart contracts have been audited and several suspicious places were found. During the audit no critical or major issues were identified. Several issues were marked as warnings and comments. After working on audit report all issues were fixed or acknowledged by client (if the problem was not critical). Thus, contracts are assumed as secure to use according to our security criteria.Final commit identifier with all fixes: `c24beb01afd7f6397e78062dd9a46906d38bc08b`

2. FINDINGS REPORT

2.1 CRITICAL

Not Found

2.2 MAJOR

Not Found

2.3 WARNING

WRN-1	Check address for zero
File	AnchorVault.vy
Severity	Warning
Status	Acknowledged

DESCRIPTION

At line: AnchorVault.vy#L108
it is possible to set admin with zero address.

RECOMMENDATION

We recommend to add following check:

```
require(new_admin!= address(0), "Incorrect address");
```

CLIENT'S COMMENTARY

This is intentional to allow ossification of the contract, i.e. making it irreversibly non-administrable. This is done via setting the admin to zero.

2.4 COMMENT

CMT-1	Use constant 100
File	AnchorVault.vy
Severity	Comment
Status	Fixed at c24beb01

DESCRIPTION

At line: AnchorVault.vy#L255
`_can_deposit_or_withdraw()` use compare difference share_prices with fix constant 100. Considering that the operation of user functions `submit()` and `withdraw()` depends on this comparison we recommend making this constant variable.

RECOMMENDATION

We recommend to add new function which can change this parameter.

CMT-2	Possible gas saving
File	RewardsLiquidator.vy
Severity	Comment
Status	Fixed at c24beb01

DESCRIPTION

There are optional checks (RewardsLiquidator.vy#L197):

```
assert ERC20Decimals(UST_TOKEN).decimals() == 18
assert ERC20Decimals(STETH_TOKEN).decimals() == 18
```

RECOMMENDATION

We recommend to add these checks in the constructor.

CMT-3	Return value
File	AnchorVault.vy
Severity	Comment
Status	Fixed at c24beb01

DESCRIPTION

At line : AnchorVault.vy#L273-L304 `submit()` function doesn't return an actual `steth_amount_adj`. So the method may be blackbox for a user.

RECOMMENDATION

We recommend to return value in `submit`.

CMT-4	Sweep tokens
File	RewardsLiquidator.vy
Severity	Comment
Status	Acknowledged

DESCRIPTION

In `RewardsLiquidator` after `liquidate` (RewardsLiquidator.vy#L191) some tokens can remain in the contract.

RECOMMENDATION

We recommend to sweep tokens in the end of `liquidate`.

CLIENT'S COMMENTARY

It doesn't seem that there might be any remainder left on the liquidator contract. stETH rounding due to conversion to shares isn't possible since we're asking Curve to `transferFrom()` the whole stETH balance of the liquidator which is guaranteed to be the whole number of shares.
Later, the whole amount of the received Ether is transferred to Sushi, and Sushi is instructed to transfer the UST result to the recipient address.

Update: actually, transferring the whole stETH balance (including selling it on Sushi) might leave some dust on the contract's address (less than 10 wei) due to the internal rounding in the stETH contract. However, this dust is not transferrable due to the same rounding so we cannot send it back to the caller.
It also cannot be exploited by e.g. transferring more stETH to the rewards liquidator contract and selling the whole amount including the dust afterwards since the transaction cost would be orders of magnitude more than the revenue from selling the dust.

3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build open-source solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

BLOCKCHAINS

- Ethereum
- Cosmos
- EOS
- Substrate

TECH STACK

- Python
- Solidity
- Rust
- C++

CONTACTS

https://github.com/mixbytes/audits_public

https://mixbytes.io/

hello@mixbytes.io

https://t.me/MixBytes

https://twitter.com/mixbytes