Lido
1inch Rewards Manager
SMART CONTRACT AUDIT

September 23, 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. 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 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.

1.3 EXECUTIVE SUMMARY

The smart contract reviewed in this audit is designed to be a mediator contract between LIDO's DAO voting and 1INCH reward farming facility. The main problem is that LIDO uses DAO votings to allocate tokens for rewards. Making a direct transfer from the DAO to the reward contract is not an option. The `RewardsManager` contract is used to calculate the time to start the voting in order to make it passed exactly by the end of the previous reward period.This smart contract has the following main functions:

- `start_next_rewards_period()` - starts the next rewards via calling `FarmingRewards.notifyRewardAmount()` and transferring `ldo_token.balanceOf(self)` tokens to `FarmingRewards`. The `FarmingRewards` contract handles all the rest on its own. The current rewards period must be finished by this time. First period could be started only by `self.rewards_initializer`. It can only be called by somebody.
- `recover_erc20()` - transfers the given _amount of the given ERC20 token from itself to the recipient. It can only be called by the owner.
- `transfer_ownership()` - changes the contract owner. It can only be called by the current owner.
- `set_rewards_period_duration()` - updates period duration. It can only be called by the owner.
- `period_finish()` - shows the end date of the voting period.
- `is_rewards_period_finished()` - shows whether the current rewards period has finished.

1.4 PROJECT DASHBOARD

Client	Lido
Audit name	1inch Rewards Manager
Initial version	c2cd9665666deda9452fa9e3461fbf3537413945
Final version	c2cd9665666deda9452fa9e3461fbf3537413945
Date	September 10, 2021 - September 23, 2021
Auditors engaged	3 auditors

FILES LISTING

RewardsManager.vy	https://github.com/lidofinance/1inch-rewards-manager/blob/c2cd9665666deda9452fa9e3461fbf3537413945/contracts/RewardsManager.vy

FINDINGS SUMMARY

Level	Amount
Critical	0
Major	0
Warning	2
Comment	2

CONCLUSION

During the audit no critical issues were found, several warnings and comments were spotted. After working on the reported findings all of them were acknowledged.Final commit identifier with all fixes: `c2cd9665666deda9452fa9e3461fbf3537413945`

2. FINDINGS REPORT

2.1 CRITICAL

Not Found

2.2 MAJOR

Not Found

2.3 WARNING

WRN-1	No check of the address parameter for zero
File	RewardsManager.vy
Severity	Warning
Status	Acknowledged

DESCRIPTION

At line RewardsManager.vy#L66, the `owner` variable is assigned the value of the `_to` parameter.
But, if by chance the value of the parameter turns out to be equal to zero, then the work of the following functions will be blocked:
`set_rewards_period_duration()`, `recover_erc20()`.

RECOMMENDATION

It is recommended to add a check for the variable `_to` for zero before line 66.

CLIENT'S COMMENTARY

> Acknowledged. Since the contract has already been deployed and is not supposed to be transferred to another owner, we consider this risk eliminated, with no changes planned.

WRN-2	No logging of important events
File	RewardsManager.vy
Severity	Warning
Status	Acknowledged

DESCRIPTION

Logging important actions makes it easier to maintain the project.
But in this smart contract it is not done for some important events.
At lines RewardsManager.vy#L100-L120 for the `start_next_rewards_period()` external function this event logging is lacking.
At lines RewardsManager.vy#L124-L131 for the external function `set_rewards_period_duration()` this event is not logged.

RECOMMENDATION

It is recommended to add logging of important events.

CLIENT'S COMMENTARY

> Acknowledged. Since rewards manager is a minor utility contract we don't expect these events to be listened to, they are also covered in the FarmingRewards contract.

2.4 COMMENT

CMT-1	No setter for the address of the interacting contract
File	RewardsManager.vy
Severity	Comment
Status	Acknowledged

DESCRIPTION

At line RewardsManager.vy#L39 the `rewards_contract` variable is described.
This is the address of the `FarmingRewards` contract. Now there is no way to change its address. If new functionality is added to this contract, it will be necessary to reinstall the `RewardsManager` contract.

RECOMMENDATION

It is recommended to add a method to change the value of the variable `rewards_contract`.

CLIENT'S COMMENTARY

> We consider this to be no issue, the behavior described is expected. In case of any significant farming program changes, the rewards manager is supposed to be redeployed.

CMT-2	Test scripts problem
File	
Severity	Comment
Status	Acknowledged

DESCRIPTION

This problem is not in the audit scope.
The following two tests do not work correctly:
`test_owner_recovers_erc20_to_own_address` and `test_owner_recovers_erc20_zero_amount`.

RECOMMENDATION

It is recommended to fix the tests.

CLIENT'S COMMENTARY

> Acknowledged. The contract has been deployed already, no fix will be done for this issue.

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