ABYSS LOCKUP SMART CONTRACT AUDIT

February 24, 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 Abyss 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

Smart contract for ERC20 and LP tokens lockups with 1, 3, 6, 12 months delay after withdrawal request.

1.3 SECURITY ASSESSMENT METHODOLOGY

At least 2 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 "Blind" audit includes:
> Manual code study
> "Reverse" research and study of the architecture of the code based on the source code only

Stage goal:
Building an independent view of the project's architecture
Finding logical flaws

02 Checking the code against the checklist of known vulnerabilities includes:
> 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

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

03 Checking the logic, architecture of the security model for compliance with the desired model, which includes:
> 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

Stage goal:
Detection of inconsistencies with the desired model

04 Consolidation of the reports from all auditors into one common interim report document
> 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
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

Audited scope includes contracts which are the part of multifunctional lockup mechanism. General purpose of contracts is allowing users to lockup any kind of ERC-20 tokens for already defined time, but e.g. contracts also can be used as staking reward distributor.

1.5 PROJECT DASHBOARD

Client	Abyss Finance
Audit name	Abyss Lockup
Initial version	8fe1a854a9b01dc1aa35272b82fd22655d4f42d1
Final version	77ce2ef196b8aee29874b7d8f1d4005a552d5c08
SLOC	467
Date	2021-02-02 - 2021-02-24
Auditors engaged	2 auditors

FILES LISTING

AbyssLockup.sol	AbyssLockup.sol
AbyssSafe1.sol	AbyssSafe1.sol
AbyssSafe3.sol	AbyssSafe3.sol
AbyssSafe6.sol	AbyssSafe6.sol
AbyssSafe12.sol	AbyssSafe12.sol
IAbyssLockup.sol	IAbyssLockup.sol

FINDINGS SUMMARY

Level	Amount
Critical	2
Major	0
Warning	1
Comment	2

CONCLUSION

Smart contracts have been audited and several suspicious places were found. During audit 2 critical were identified as they could lead to wrong behavior related with user assets and several issues were marked as warning or comment. After working on audit report all issues were fixed or acknowledged by client and contracts assumed as secure to use according our security criteria. Final commit identifier with all fixes: `77ce2ef196b8aee29874b7d8f1d4005a552d5c08`.

2. FINDINGS REPORT

2.1 CRITICAL

CRT-1	Unfair withdrawn amount
File	https://gist.github.com/algys/eb905ec8efa41f80cf1eab57a3b31649
Severity	Critical
Status	Fixed at edd0cb49

DESCRIPTION

How to reproduce bug:

- Deposit N tokens from Alice
- Deposit N tokens from Bob
- Deposit N tokens from Eve
- Request and withdraw N tokens to Alice
- Request and withdraw N tokens to Bob
- Request and withdraw N tokens to Eve
- At this point participant got different withdrawn amount(first lost more funds)

Detailed explanation:

- Use particular deflationary token as depositing asset https://gist.github.com/algys/eb905ec8efa41f80cf1eab57a3b31649
- After all withdrawals Alice lost more funds than Eve, that behavior is unfair because they deposited same amount and just lost funds depending on withdrawal order

RECOMMENDATION

It is recommended to refactor rebase logic and reduce amount of code duplication.

CRT-2	Potential withdrawal lock and invalid distribution
File	https://gist.github.com/algys/eb905ec8efa41f80cf1eab57a3b31649
Severity	Critical
Status	Fixed at **77ce2ef1**

DESCRIPTION

How to reproduce bug:

- Deposit N tokens from Alice
- Deposit N tokens from Bob
- Deposit N tokens from Eve
- Send M (relatively huge amount) to Lockup contract directly (via transfer)
- Request and withdraw N tokens to Alice
- Request and withdraw N tokens to Bob
- Request and withdraw N tokens to Eve
- At this point participant got different withdrawn amount(first lost more funds), and depending on M amount sometimes contract can be failed on `request` call

Detailed explanation:

- Use particular deflationary token as depositing asset https://gist.github.com/algys/eb905ec8efa41f80cf1eab57a3b31649
- After all withdrawals Alice lost more funds than Eve, that behavior is unfair because they deposited same amount and just lost funds depending on withdrawal order

RECOMMENDATION

It is recommended to fix rebase logic related to lockup balance based calculation

2.2 MAJOR

Not Found

2.3 WARNING

WRN-1	Potentially `approved` cache miss
File	AbyssSafe3.sol
Severity	Warning
Status	Fixed at 816328e8

DESCRIPTION

At lines AbyssSafe3.sol#L202-L216 contract has approval caching mechanism, that works fine if token supports infinity approval, so in other cases cached approve might tell wrong info.

RECOMMENDATION

It is recommended to check approval permanently

2.4 COMMENTS

CMT-1	Use simplified syntax while working with libs
File	AbyssSafe3.sol
Severity	Comment
Status	Fixed at ad710b12

DESCRIPTION

In reviewed contracts e.g file AbyssSafe3.sol explicit syntax used everywhere while working with SafeMath lib:

```
SafeMath.div(
  SafeMath.mul(
      _data[msg.sender][token].deposited,
      _tokens[token].divFactorDeposited
      ),
  _data[msg.sender][token].divFactorDeposited
  );
```

it's better to use simplified one:

```
_data[msg.sender]
[token].deposited.mul(_tokens[token].divFactorDeposited).div(_data[msg.sender]
[token].divFactorDeposited)
```

RECOMMENDATION

It is recommended to use simplified syntax

CMT-2	Reduce copy-pasted code amount
File	AbyssSafe3.sol
Severity	Comment
Status	**Fixed** at **d98756f6**

DESCRIPTION

There are a lot of places when rebases' logic code is copy-pasted:

- AbyssSafe3.sol#L223-L239
- AbyssSafe3.sol#L330-L344
- AbyssSafe3.sol#L472-L482

Code duplication highly increases probability to introducing bugs and makes code reading and reviewing process really hard that increase bug missing probability as well

RECOMMENDATION

It is recommended to incapsulate similar code to special function

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