ORBXBALL STABLECOINS 3POOL SMART CONTRACT AUDIT

March 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 Yearn. 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

Basic Solidity Smart Contract for creating your own Yearn Strategy.

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

The audited scope includes three implementations of the BaseStrategy strategy for tokens: DAI, USDC, USDT. The strategies implemented profit management rules that the user will receive after depositing their tokens in vaults for each token. Interestingly, the strategy for each token depends on two other tokens to work.

1.5 PROJECT DASHBOARD

Client	Yearn
Audit name	Stablecoins-3pool
Initial version	adeb776933c6cb3b8306239cc3357d4c6239a88d 3f772afc4f868cc2ec9112a86b0cb89a838a7ae8 efbf10ec8d2c624dfcf4685cf905220989716f55 e86260fa8f786bcf7f39503ec80250afd144366a
Final version	e86260fa8f786bcf7f39503ec80250afd144366a
SLOC	566
Date	2021-03-10 - 2021-03-24
Auditors engaged	2 auditors

FILES LISTING

StrategyDAI.sol	StrategyDAI.sol
StrategyUSDC.sol	StrategyUSDC.sol
StrategyUSDT.sol	StrategyUSDT.sol

FINDINGS SUMMARY

Level	Amount
Critical	0
Major	2
Warning	6
Comment	2

CONCLUSION

Smart contracts have been audited and several suspicious places have been spotted. During the audit no critical issues were found, two issues was marked as major because it could lead to some undesired behavior, also several warnings and comments were found and fixed by the client. After working on the reported findings all of them were resolved or acknowledged (if the problem was not critical). So, the contracts are assumed as secure to use according to our security criteria.Final commit identifier with all fixes: `e86260fa8f786bcf7f39503ec80250afd144366a`

2. FINDINGS REPORT

2.1 CRITICAL

Not Found

2.2 MAJOR

MJR-1	Withdrawal from an account exceeding availible assets
File	StrategyDAI.sol StrategyUSDC.sol
Severity	Major
Status	Fixed at 3f772afc

DESCRIPTION

The contractor discovered a code that is potentially susceptible to contract failure due to throw from another contract.

- StrategyDAI.sol#L164
- StrategyUSDC.sol#L164

RECOMMENDATION

It is recommended to add validation which was done in the contract:

- StrategyUSDT.sol#L163

CLIENT'S COMMENTARY

> no comment

MJR-2	Incorrect transfer of parameter values
File	StrategyUSDC.sol StrategyUSDT.sol
Severity	Major
Status	Fixed at efbf10ec

DESCRIPTION

At the lines:

- StrategyUSDC.sol#L222
- StrategyUSDT.sol#L224
  the parameters are not passed correctly.

RECOMMENDATION

It is recommended to fixed it.

CLIENT'S COMMENTARY

> no comment

2.3 WARNING

WRN-1	Forced deposit from an account exceeding available assets
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Fixed at e86260fa

DESCRIPTION

It is possible get an exception due to throw from another contract while depositing.

- StrategyDAI.sol#L207
- StrategyUSDC.sol#L207
- StrategyUSDT.sol#L209

RECOMMENDATION

It is recommended to add the following check:

```
uint256 _amt = want.balanceOf(address(this));
if (_amount > _amt) _amount = _amt;
```

CLIENT'S COMMENTARY

> no comment

WRN-2	Forced withdrawal from an account exceeding available assets
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Fixed at e86260fa

DESCRIPTION

Contract potentially can failure due to throw from another contract while withdrawal amount of tokens which exceed available assets.

- StrategyDAI.sol#L217
- StrategyUSDC.sol#L217
- StrategyUSDT.sol#L219

RECOMMENDATION

A possible solution would be to add the following check:

```
uint256 _bal = IERC20(y3crv).balanceOf(address(this));
if (_amt > _bal) _amt = _bal;
```

CLIENT'S COMMENTARY

> no comment

WRN-3	There is no input parameter processing in the method
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

At the lines:

- StrategyDAI.sol#L99
- StrategyUSDC.sol#L99
- StrategyUSDT.sol#L99
  the `adjustPosition()` method has an input variable `_debtOutstanding`.
  But there is no processing of this variable in the body of the function.

RECOMMENDATION

It is recommended to either add handling to the variable or remove this variable.

CLIENT'S COMMENTARY

> no comment

WRN-4	The approval value obtained in the constructor may not be enough for the long term of the smart contract
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Smart contracts call `safeApprove()` functions for different tokens. But in the process of work, the obtained value will only decrease. If this value decreases to zero, then the tokens will remain locked in the contract forever.
It is at the following lines:

- StrategyDAI.sol#L39-L41
- StrategyUSDC.sol#L39-L41
- StrategyUSDT.sol#L39-L41

RECOMMENDATION

It is recommended to add a function to increase the value of approvals.

CLIENT'S COMMENTARY

> no comment

WRN-5	Possible flashloan attacks
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

In current contract some functions can be influenced by flashloan attacks.

- StrategyDAI.sol#L71
- StrategyUSDC.sol#L71
- StrategyUSDT.sol#L71

RECOMMENDATION

It is recommended to check the return value for validity when calling the `balanceOfy3CRVinWant()` function.

CLIENT'S COMMENTARY

> We have a `maxDepositAmount`, which can limit the max deposit amount on this strategy. Also, from the vault side it controls the influx and efflux of big deposit/withdraw amount to avoid that amount directly interact with this strategy. and the third protection is the tight slippage.

WRN-6	Correct migration
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

In constructor, rights are granted to spend tokens, which should be canceled when migrating the strategy.

- StrategyDAI.sol#L175
- StrategyUSDC.sol#L175
- StrategyUSDT.sol#L177

RECOMMENDATION

It is recommended to add in function `prepareMigration()`:

```
want.safeTransfer(_newStrategy, want.balanceOf(address(this)));
want.safeApprove(_3pool, 0);
IERC20(_3crv).safeApprove(y3crv, 0);
IERC20(_3crv).safeApprove(_3pool, 0);
```

CLIENT'S COMMENTARY

> no comment

2.4 COMMENTS

CMT-1	Event is probably missing
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

At the lines:

- StrategyDAI.sol#L205-L213
- StrategyUSDC.sol#L205-L213
- StrategyUSDT.sol#L207-L215

in method `forceD()` should probably emit an event `ForceDeposit`.

At the lines:

- StrategyDAI.sol#L215-L225
- StrategyUSDC.sol#L215-L225
- StrategyUSDT.sol#L217-L227

in method `forceW()` should probably emit an event `ForceWithdraw`.

At the lines:

- StrategyDAI.sol#L44-L50
- StrategyUSDC.sol#L44-L50
- StrategyUSDT.sol#L44-L50

the methods `setThreshold()` and `setSlip` should probably emit an events: `NewThreshold` and `NewSlip`.

At the lines:

- StrategyDAI.sol#L197-L203
- StrategyUSDC.sol#L197-L203
- StrategyUSDT.sol#L199-L205

in method `rebalance()` should probably emit an event `Rebalance`.

RECOMMENDATION

It is recommended to create new events.

CLIENT'S COMMENTARY

> no comment

CMT-2	Not informative names of functions and variables
File	StrategyDAI.sol StrategyUSDC.sol StrategyUSDT.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

For the function names `forceD ()` and `forceW ()`, it is not clear what these functions are used for.
It is at the following lines:

- StrategyDAI.sol#L205-L225
- StrategyUSDC.sol#L205-L225
- StrategyUSDT.sol#L207-L227

For the names of the variables `p` and `_p`, it is impossible to understand what these variables are used for.
It is on the following lines:

- StrategyDAI.sol#L84-L92
- StrategyUSDC.sol#L84-L92
- StrategyUSDT.sol#L84-L92

Correct names of functions and variables make programs easier to use.

RECOMMENDATION

It is recommended to give correct names to functions and variables.

CLIENT'S COMMENTARY

> no comment

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