YEARN MAKER DAI DELEGATE SMART CONTRACT AUDIT

October 18, 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 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 PROJECT OVERVIEW

The maker DAI delegate project is Yearn vault strategy to earn a profit investing non-DAI ERC20 token into Yearn DAI vault without swapping non-DAI to DAI by using non-DAI as a collateral for minting DAI.Like other Yearn strategies, the implementation consists of the base class `BaseStrategy` extended by custom logic `Strategy`, implemented in `BaseStrategy.sol` and `Strategy.sol`, respectively.The strategy maintains changes of collateral price to invest as much DAI as possible to avoid liquidation risks.

1.4 PROJECT DASHBOARD

Client	Yearn
Audit name	Maker Dai Delegate
Initial version	97949a51062df956fd0172b7b1c778f66844b634
Final version	97949a51062df956fd0172b7b1c778f66844b634
Date	September 13, 2021 - October 18, 2021
Auditors engaged	3 auditors

FILES LISTING

MakerDaiDelegateCloner.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/contracts/MakerDaiDelegateCloner.sol
Strategy.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/contracts/Strategy.sol
TestStrategy.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/contracts/TestStrategy.sol
MakerDaiDelegateLib.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/contracts/libraries/MakerDaiDelegateLib.sol
AggregatorInterface.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/interfaces/chainlink/AggregatorInterface.sol

IMaker.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/interfaces/maker/IMaker.sol
ISwap.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/interfaces/swap/ISwap.sol
IOSMedianizer.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/interfaces/yearn/IOSMedianizer.sol
IVault.sol	https://github.com/therealmonoloco/maker-dai-delegate/blob/97949a51062df956fd0172b7b1c778f66844b634/interfaces/yearn/IVault.sol
BaseStrategy.sol	https://github.com/gzliudan/yearn-vaults/blob/b626994c5ca2ed4455053809cdd5b75bd567ccbf/contracts/BaseStrategy.sol

FINDINGS SUMMARY

Level	Amount
Critical	0
Major	2
Warning	0
Comment	2

CONCLUSION

During the audit process, several smart contracts and its interaction were examined. In some extremely rare conditions, several issues may potentially occur. These issues can be fixed by manual intervention of the Yearn Governance and may not lead to permanent losses of users' funds.Assuming the Yearn Governance is trusted, no potential risks for users' funds were found.Final commit identifier with all fixes: `97949a51062df956fd0172b7b1c778f66844b634`

2. FINDINGS REPORT

2.1 CRITICAL

Not Found

2.2 MAJOR

MJR-1	Improper handling of losses in Yearn DAI vault
File	Strategy.sol
Severity	Major
Status	Acknowledged

DESCRIPTION

The strategy is using another Yearn vault to invest DAI to take profit from it. In some rare conditions, that vault can suffer losses. Such conditions are not properly handled.

1. At the Strategy.sol#L303 the strategy is not reporting losses from `DAI yVault` until it is completely unable to conceal. As a result, losses are not fairly distributed between vault users. Some users may suffer unfair penalty on withdraw.
2. At the Strategy.sol#L407 in some conditions, the strategy can overreport losses while it still has enought assets to complete the liquidation request by rebalancing `DAI` debt/collateral at the `Maker vault`.
3. The losses in `DAI yVault` are never fixed, so even newcoming investors will not be able to take a profit until recent losses are compensated. On large losses it can take years to recover.

RECOMMENDATION

It is recommended to implement manual function to fix losses at `DAI yVault`:

1. To report losses to the parent vault
2. To rebalance debt/collateral at Maker vault to make DAI debt equals to DAI balance available to the strategy

In some rare conditions, the option to flash-borrow WANT tokens from "the fixer" can be required to proceed rebalance during a liquidity shortage.

CLIENT'S COMMENTARY

> 1. This is a design decision we take for debt-based strategies. Losses are not reported as they go but only at the time they are realized.
> 2. This can still be handled by governance by granting CDP managing rights to itself with grantCdpManagingRightsToUser - which would allow raw / direct access to the CDP (repay debt, free collateral), or doing a WANT airdrop to the strategy
> 3. On extreme scenarios with unrecoverable losses we will most likely migrate to a new vault or amend losses by creating additional profit via an airdrop to the strategy

MJR-2	Strategy migration reverts after Maker liquidation
File	Strategy.sol
Severity	Major
Status	Acknowledged

DESCRIPTION

Unlikely, but possible situation when `Maker` collateral will be liquidated. In this case all operations with collateral will be reverted because the owner would be changed. After the liquidation you can't migrate to a new strategy because `prepareMigration` will be reverted for the above reasons.

Strategy.sol#L450

RECOMMENDATION

The `Maker`'s contract does not provide any functionality for getting the collateral's owner by cdpId (for checking that the liquidation didn't happen). That's why you have to cautch all exceptions of transferCdp call by using try/catch functionality in Solidity (0.6 and higher) and check the security modifier error.

CLIENT'S COMMENTARY

Remaining yvDAI shares can be easily migrated by governance using `migrateToNewDaiYVault` function

2.3 WARNING

Not Found

2.4 COMMENT

CMT-1	Unnecessary assert on immutable variable
File	Strategy.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Assertion `address(investmentToken) != address(want)` in `_buyInvestmentTokenWithWant` is not necessary during contract execution

Strategy.sol#L855

RECOMMENDATION

It is recommended to make this assertion in `initialize` function

CLIENT'S COMMENTARY

Reason to keep first one as-is is mostly due to shared code between debt-based strategies and preventing introduction of bugs in case another strategist copy-pastes for a strategy that needs the additional check.

CMT-2	Gas optimization in `_convertInvestmentTokenToWant`
File	Strategy.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Two of three calls to `_convertInvestmentTokenToWant` can be saved at Strategy.sol#L282 by regrouping the add/mul operations.

RECOMMENDATION

We recommend to regroup the add/mul operations.

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