CONVEX PLATFORM SMART CONTRACT AUDIT

April 19, 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 Convex. 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

Convex Platform implies community based staking with boosting without the need for locking yourself.

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 contains smart contract of convex platform project. The main project's goal is automation and boosting rewards from curve gauges. Users can deposit their curve LP tokens to convex pool, pool automatically locks it into gauges and get reward in crv token, crv also can be locked in curve to gain additional reward from curve booster.

1.5 PROJECT DASHBOARD

Client	Convex
Audit name	Convex Platform
Initial version	754d9e700693246275b613e895b4044b63ce9ed5
Final version	0c61de7461124d9124384574e1017e55c01607bf
SLOC	2105
Date	2021-03-15 - 2021-04-19
Auditors engaged	2 auditors

FILES LISTING

VoterProxy.sol	VoterProxy.sol
BaseRewardPool.sol	BaseRewardPool.sol
CrvDepositor.sol	CrvDepositor.sol
Interfaces.sol	Interfaces.sol
StashFactory.sol	StashFactory.sol
DepositToken.sol	DepositToken.sol
Cvx.sol	Cvx.sol
ExtraRewardStashV2.sol	ExtraRewardStashV2.sol
Booster.sol	Booster.sol
ManagedRewardPool.sol	ManagedRewardPool.sol
RewardFactory.sol	RewardFactory.sol
cCrv.sol	cCrv.sol
DebugInterfaces.sol	DebugInterfaces.sol
cCrvRewardPool.sol	cCrvRewardPool.sol
TokenFactory.sol	TokenFactory.sol
ExtraRewardStashV1.sol	ExtraRewardStashV1.sol
cvxRewardPool.sol	cvxRewardPool.sol
VirtualBalanceRewardPool.sol	VirtualBalanceRewardP...

FINDINGS SUMMARY

Level	Amount
Critical	1
Major	3
Warning	5
Comment	8

CONCLUSION

Smart contract have been audited and several suspicious places have been spotted. During the audit 1 critical and 3 major issues were found, also several warnings and comments were found and included to report. After working on the reported findings all of them were resolved or acknowledged (if the problem was not critical). Final commit identifier with all fixes: `0c61de7461124d9124384574e1017e55c01607bf`

2.FINDINGS REPORT

2.1 CRITICAL

CRT-1	Anyone can perform any arbitrary calls on behalf of `VoterProxy`
File	VoterProxy.sol
Severity	Critical
Status	Fixed at **ef433b15**

DESCRIPTION

Function `deposit` in `VoterProxy` defined at VoterProxy.sol#L60 accepts call from anyone with any `_token` and `_gauge`, so anyone can craft calldata and make call on behalf of `VoterProxy`. That could lead to undesired behavior, e.g user's funds locked in contract or authorization violation. Also deposit can be called for `_gauge` and `_token` which are not compatible.
Moreover for now anyone can allow spending tokens from contract balance for any third-party account by calling deposit for target token with evil gauge.

RECOMMENDATION

We strictly recommend to whitelist `_gauge` and `_token`. And also check that `_token` and `_gauge` are compatible(check that `ICurveGauge(_gauge).withdraw(_amount)` returns right `_token`)

2.2 MAJOR

MJR-1	Unstable gauge version check
File	StashFactory.sol
Severity	Major
Status	Fixed at 1858521a

DESCRIPTION

`ShashFactory` contract have gauge version check based on call probes defined at StashFactory.sol#L51-L61, that approach is very dangerous in case of new version added to curve. E.g if curve will add new version of gauge that have `rewarded_token()` or `reward_tokens(uint256)` and with different behavior, then version checker will wrongly classify version and allow to create stash with invalid version. That can lead to broken logic.

RECOMMENDATION

We recommend to use another approach to check version, e.g whitelisting gauges. Curve have only around ~40 gauges.

MJR-2	Wrong logic in `withdrawAll`
File	VoterProxy.sol
Severity	Major
Status	Fixed at **ef433b15**

DESCRIPTION

At the moment `withdrawAll` counts balance as: `balanceOfPool(_gauge)` (VoterProxy.sol#L92)

Correct logic should be: `balanceOfPool(_gauge).add(IERC20(_token).balanceOf(address(this)))`.

The `withdrawAll` method is used by `shutdownSystem` so potentially some tokens could remain in the contract.

RECOMMENDATION

It is recommended to count amount of tokens as `balanceOfPool(_gauge).add(IERC20(_token).balanceOf(address(this)))`.

MJR-3	Zero gauge could be added via `addPool`
File	Booster.sol StashFactory.sol
Severity	Major
Status	Fixed at 1858521a

DESCRIPTION

In `addPool` defined at Booster.sol#L160 there is no check for `_gauge` variable. For example during this call

```
booster.addPool(threeCrvSwap, "0x0000000000000000000000000000000000000000", 0)
```

Gauge will be found because `get_gauges` returns array like `[address1, address2, 0x0, 0x0, ...]`. Intruder can call some errors in `Booster` logic.

It's major because at the moment `StashFactory` call `address(0x0).call.value(0)(data)` (StashFactory.sol#L53) and due to specific of EVM there is `true`.

RECOMMENDATION

We recommend to add some checks for `_gauge` variable.

2.3 WARNING

WRN-1	Inconsistent minted and deposited LP tokens amount
File	Booster.sol
Severity	Warning
Status	**Fixed** at **c1779fa7**

DESCRIPTION

Function `deposit` in `Booster` defined at Booster.sol#L275 allows to deposit curve pools LP token and mint wrapped convex tokens with 1:1 proportions. However minted tokens amount for user can be different from deposited LP tokens amount:

- At line Booster.sol#L278 contract accepts `_amount` LP tokens
- At line Booster.sol#L265 contract deposit `bal` tokens to gauge
- `bal != _amount` if before user deposit someone send LP token directly to `Booster` contract, so here we got that amount of deposited tokens to gauge not equal to LP tokens amount deposited to `Booster`

RECOMMENDATION

We recommend to pass actual deposited `_amount` to `sendTokensToGauge` function and use it as amount of tokens for depositing to gauge.

WRN-2	voteDelegate can perform any arbitrary calls on behalf of VoterProxy
File	VoterProxy.sol
Severity	Warning
Status	Fixed at ffb814d7

DESCRIPTION

Function `vote` in `VoterProxy` defined at VoterProxy.sol#L130 accepts call from `voteDelegate` through `Booster` contract and can call any arbitrary contract on behalf of `VoterProxy`. Since `VoterProxy` is main contract that holds users money it's highly risked to allow arbitrary contracts calls.

RECOMMENDATION

We strictly recommend to whitelist `_votingAddress`

WRN-3	Insecure privileges for `Owner`
File	Booster.sol
Severity	Warning
Status	Fixed at **b0f9b09d**

DESCRIPTION

Owner can change factories in `setFactories` (Booster.sol#L95):

```
rewardFactory = _rfactory;
stashFactory = _sfactory;
tokenFactory = _tfactory;
```

Via front-running attack `owner` can change these addresses before calling `addPool`.

RECOMMENDATION

We recommend to construct this contracts in Booster and create mechanism of migrations directly in factories.

WRN-4	Call `earmarkRewards` after shutdown
File	Booster.sol
Severity	Warning
Status	Fixed at **fb25f601**

DESCRIPTION

Line is commented in method `earmarkRewards` defined at Booster.sol#L434

```
// require(!isShutdown,"shutdown");
```

However if system is shutdowned the transaction would be reverted because stash has no access to `VoterProxy`.

RECOMMENDATION

It is recommended to uncomment this line.

WRN-5	Missed `safeApprove`
File	Booster.sol
Severity	Warning
Status	Fixed at 19d58143, 0c61de74

DESCRIPTION

`Booster` uses `approve` method (Booster.sol#L288):

```
IERC20(token).approve(rewardContract, _amount);
```

It's better to use `safeApprove`.

RECOMMENDATION

It is recommended to use `safeApprove`.

2.4 COMMENTS

CMT-1	Cache `poolInfo` in memory to save gas
File	Booster.sol
Severity	Comment
Status	Fixed at 7cd1773e, 64b8c045

DESCRIPTION

In function `deposit` of `Booster` contract defined at Booster.sol#L275 there are several reads of `poolInfo` struct fields, so it's better to cache `poolInfo` structure in memory to save some gas on reading.

RECOMMENDATION

We suggest to cache `poolInfo` in memory

CMT-2	Check user balance at beginning to save gas
File	Booster.sol
Severity	Comment
Status	Fixed at 7cd1773e, 64b8c045

DESCRIPTION

Function `_withdraw` defined at line Booster.sol#L309 needs to burn wrapper tokens and back LP tokens to user, for now in case if user have to sufficient wrapped tokens `ITokenMinter(token).burn(_from,_amount)` at line Booster.sol#L329 will revert transaction. In that case user will pay gas for whole operations before, so we recommend to check user's balance at the very beginning of the functions to save gas on negative scenario.

RECOMMENDATION

We recommend to check user's balance at beginning of the function

CMT-3	Remove unrelevant commentaries
File	Booster.sol
Severity	Comment
Status	**Fixed** at **8d9e0eab**

DESCRIPTION

At lines:

- Booster.sol#L345
- Booster.sol#L441
- Booster.sol#L119
- etc

there are commentaries which are not really relevant

RECOMMENDATION

We recommend to remove unneeded comments

CMT-4	Reduce amount of code duplication
File	BaseRewardPool.sol ManagedRewardPool.sol VirtualBalanceRewardPool.sol cCrvRewardPool.sol cvxRewardPool.sol
Severity	Comment
Status	Fixed at 07280159

DESCRIPTION

Contracts:

- BaseRewardPool.sol
- ManagedRewardPool.sol
- VirtualBalanceRewardPool.sol
- cCrvRewardPool.sol
- cvxRewardPool.sol

have a lot of intersections in terms of code duplication, so it's bad practice because it makes easier to introduce bug and makes code more complex

RECOMMENDATION

We recommend to reduce duplication using contracts inheritance

CMT-5	Confusing naming of subjects
File	VoterProxy.sol Booster.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

At several places there are confusing naming, e.g:

- VoterProxy.sol#L23 `operator` is `Booster`
- Booster.sol#L62 `staker` is `VoterProxy`
- etc

it's always better to have strict, unambiguous and transparent naming, same things should have same names through whole project to make project more readable and simpler.

RECOMMENDATION

We recommend use unambiguous naming in whole project.

CMT-6	Confusing interfaces
File	Interfaces.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

There are a lot of interfaces in file Interfaces.sol, some of that interfaces used in project and sometimes it's not clear what interface is internal(interface of project contract) and what interface is external(e.g curve's one)

RECOMMENDATION

We recommend to separate external\internal interfaces. And also recommend to keep widely used structure and naming of interfaces: contract interface should have all public methods and should be name should be like I{contract name}.sol. And interfaces should be located at 'interface' directory.

CMT-7	Saving gas while `platformFee` transferring
File	Booster.sol
Severity	Comment
Status	**Fixed** at **07c6e026**

DESCRIPTION

If `platformFee` is zero then it will call empty `safeTransfer`.

At the moment there is only one condition at line Booster.sol#L405

```
treasury != address(0) && treasury != address(this)
```

RECOMMENDATION

We recommend to add `platformFee > 0`.

CMT-8	Check if system shutdowned in `addPool`
File	Booster.sol
Severity	Comment
Status	**Fixed** at **f36d093e**

DESCRIPTION

Method `addPool` defined at Booster.sol#L160 doesn't have checks for `isShutdown`.

RECOMMENDATION

We recommend to prevent `addPool` when system is shutdown.

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
- EOS
- Cosmos
- Substrate

TECH STACK

- Python
- Rust
- Solidity
- C++

CONTACTS

- https://github.com/mixbytes/audits_public
- https://mixbytes.io/
- hello@mixbytes.io
- https://t.me/MixBytes
- https://twitter.com/mixbytes