CERTIK

Security Assessment

StakeWithUs: Vault V2

Jul 23rd, 2021

Table of Contents

Summary

This report has been prepared for StakeWithUs to discover issues and vulnerabilities in the source code of the StakeWithUs: Vault V2 project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Majority of the findings are of informational nature with 7 minor findings. The minor findings comprise lack of validation for function parameters, ineffectual removal of token approval from dex protocols, volatile conditional statement when leveraging in `StrategyCompLev` and lack of validation for the sufficiency of Ether balance when forwarding them in TimeLock contract. The team responded to all of the findings by either remediating or declining the finding.

Overview

Project Summary

Project Name	StakeWithUs: Vault V2
Description	The report represents audit of Strategy contracts that allow users to deposit funds that are then deposited in yield farming protocols of Compound and Protocol and the profits earned on strategies are sent to their respective `fundManager` contracts.
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/stakewithus/unagii-vault-v2/tree/d1af693b837774c11c26ba930efc2c16f9a3346b/contracts https://github.com/stakewithus/unagii-vault-v2/commit/0cdc6074ac49797b3d5a30d5243caefd29fb0563
Commit	d1af693b837774c11c26ba930efc2c16f9a3346b 0cdc6074ac49797b3d5a30d5243caefd29fb0563

Audit Summary

Delivery Date	Jul 23, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

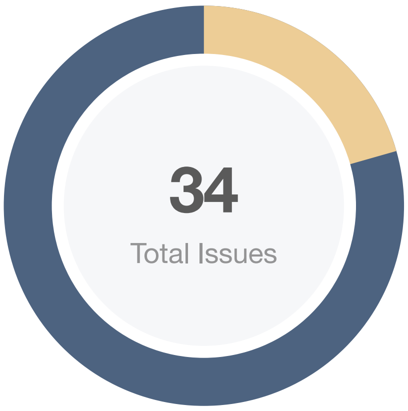
Vulnerability Level	Total	Pending	Partially Resolved	Resolved	Acknowledged	Declined
● Critical	0	0	0	0	0	0
● Major	0	0	0	0	0	0
● Medium	0	0	0	0	0	0
● Minor	7	0	0	6	0	1
● Informational	27	0	0	15	5	7
● Discussion	0	0	0	0	0	0

Audit Scope

ID	file	SHA256 Checksum
CES	interfaces/compound/CErc20.sol	ea4a0be0f26286ad90486fe1d67df4e8ac6b5e245d57b10a2240b0abdc286660
COP	interfaces/compound/Comptroller.sol	19e60b4cd2c3dcc459684840411c5bedf1c6511ab712dc28648daed360955cde
BRP	interfaces/convex/BaseRewardPool.sol	f2aaa46d04bdaabc1b82468ee11405fcc2f3dec52355a75f5f8ac0b19ea7415b
BOO	interfaces/convex/Booster.sol	0a625d4c55c93771e31e7f9f7c4362893f7aa0a716e833606438eb859f126bb5
DBS	interfaces/curve/DepositBbtc.sol	6e78d2324aa17ee777ee20147cac307badabbbd646f4de494ec4cf2f3cf8d5e1
DOS	interfaces/curve/DepositObtc.sol	aae3a9b3f59d82f2b3ea2322378f93cf40d989387539acfd5e0808e1c2f1b288
DZA	interfaces/curve/DepositZapAlUsd3Crv.sol	f929d2f9580c513f91bc8389f0c76f442414ae8c3e1084ec4d2b25f5c8ee1926
DZU	interfaces/curve/DepositZapUsdp3Crv.sol	26b25afa7249c1f82c8e1be860ea8d74a9f8f628f5c5731dad5699fe757cd3d3
SSC	interfaces/curve/StableSwap3Crv.sol	3d40de57c645f3c86f3882134038f098a94eb9adab1fece3d66ea2c13b854ebf
SSA	interfaces/curve/StableSwapAlUsd3Crv.sol	eb6697e76c9894b9654be4d5d9028d18e7f5374b3a8d1a57a92e292d5ae3cbf8
SSB	interfaces/curve/StableSwapBbtc.sol	787e0d2fe1c91e6a9df98b60e3170bb5b817fb70740266d22da5a1c175e76b9a
SSO	interfaces/curve/StableSwapObtc.sol	19a488709a3f94f99a22c63d87899010fc7b2bbef37a1b80eeaaabed96ec71bc
SSS	interfaces/curve/StableSwapSbtc.sol	22cfa2af7c4ed6b73f4161bb11cd4169101a68dd30e6158403fdcf02ab8a985a
SSE	interfaces/curve/StableSwapStEth.sol	e7e94f5554231dcc6c0a6678740aae99434fa7cefdc4ae555b7e568ba1de21e4
SSU	interfaces/curve/StableSwapUsdp3Crv.sol	e5b78b4df3e5a75733222dd08bf4f51ef21994cd8725ef68c06a2402a303a09c
UVR	interfaces/uniswap/UniswapV2Router.sol	dd70586d78b3da70e970332330472ba8b2a254247f04b5ddc849d24e02d375a0
IEF	interfaces/IEthFundManager.sol	b03d5c707926fb23b1c694160bd56b8cd4a47ea6675d19364213fd6cbd1f419a
IFM	interfaces/IFundManager.sol	818d705bac2488911ff876c9977217fdba70219e05db908ae1d1f4e303e7039d
SCL	strategies/StrategyCompLev.sol	c877959be4d95a3b08980dc22be97e41e8dd5719d64f41a14c8fe6d06d0dbf8f
SCD	strategies/StrategyCompLevDai.sol	1ef8bfef60f0b497fbfb7a929a2742627b1bcfc133912e229f10405b404873d1
SCU	strategies/StrategyCompLevUsdc.sol	1e8ec2978df02bd67d70d14a7184c3e6e3404ca342c72115b2ed8f428ada49e1
SCW	strategies/StrategyCompLevWbtc.sol	ed4ee2288368143bab00a7c713a6a0e34af1f54e30b77dbb5150a8ed1a339f2b

ID	file	SHA256 Checksum
SCA	strategies/StrategyConvexAlUsd.sol	47c55542d9ed8f97b2dedd83957a05081fc4d6bf85eba3d89f6ecac08eb4d756
SAU	strategies/StrategyConvexAlUsdDai.sol	217d6ca34bb8737c311242c2b6d3081f111b0ed96eabb071aaaf23c1a3a2234e
SUU	strategies/StrategyConvexAlUsdUsdc.sol	40ccbc1aef0c57b151f78193a892e6a199b7c30d2087f8fb850ac07e88fae945
CAU	strategies/StrategyConvexAlUsdUsdt.sol	b5cdf2f6c624694b06e3b193cf9c3b6bc6be6701020aa60a2c62b00622f2c4bd
SCB	strategies/StrategyConvexBbtc.sol	a9af9b3f5a7b5c97695db10f9724e7fdc07f872002001a4163b864e88752ec79
SBW	strategies/StrategyConvexBbtcWbtc.sol	27adc86bb0653cbad7523843d77c9a64554f1fe6ea2cefed30ee402fc89d17b6
SCS	strategies/StrategyConvexStEth.sol	de049307f3d36d1aaa69d6c9372bb4ec084190dc5e217a27ab21a41f6f3abbeb
STA	strategies/StrategyConvexUsdp.sol	4a0bcf25078647a3db77f73b66744b9535b098f03c23348e4b157854f997152e
SUD	strategies/StrategyConvexUsdpDai.sol	6003211d6c9481dccbc7ae196a894383d3b679a0fbf4cada79dfad1904250100
CUU	strategies/StrategyConvexUsdpUsdc.sol	af814d432a8b3f91add428787562a7fa0a45076193b6379bfd8d380376a79c0f
STT	strategies/StrategyConvexUsdpUsdt.sol	051e67d5d9fdfd923a713c2c2dcfa545f194ed572a74ca1539431a64699e00f5
STR	Strategy.sol	aa22742b6dcefee1f57ba76f1d96b21827fdf41c0ceec09f2a5ab0fd4211bbab
SES	StrategyEth.sol	6bfaa70da43cbad245e542a7176be6171aed95014bff63e167901a2222c5a690
TLS	TimeLock.sol	53451c881e4f39188c119714be89b2b533b0ff09e348cb18e602c982a1f7fc39

Findings



🟥	Critical	**0** (0.00%)
🟧	Major	**0** (0.00%)
🟨	Medium	**0** (0.00%)
🟨	Minor	**7** (20.59%)
🟦	Informational	**27** (79.41%)
🟩	Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
SCA-01	Admin can change dex address	**Centralization / Privilege**	● **Informational**	ⓘ **Acknowledged**
SCA-02	Rewards are not claimed and transferred in migration of strategy	Volatile Code	● Informational	✓ Resolved
SCA-03	Usage of literal for arrays' lengths	Coding Style	● Informational	✓ Resolved
SCA-04	Inefficient storage read	Gas Optimization	● Informational	⊗ Declined
SCA-05	Explicitly returning local variable	Gas Optimization	● Informational	✓ Resolved
SCB-01	Admin can change dex address	**Centralization / Privilege**	● **Informational**	ⓘ **Acknowledged**
SCB-02	Rewards are not claimed and transferred in migration of strategy	Volatile Code	● Informational	✓ Resolved
SCB-03	Usage of literal for arrays' lengths	Coding Style	● Informational	✓ Resolved
SCB-04	Inefficient storage read	Gas Optimization	● Informational	⊗ Declined
SCB-05	Explicitly returning local variable	Gas Optimization	● Informational	✓ Resolved
SCL-01	Token approval is removed from wrong address	Logical Issue	● Minor	✓ Resolved
SCL-02	Admin can change dex address	**Centralization / Privilege**	● **Informational**	ⓘ **Acknowledged**
SCL-03	Incorrect conditional	Logical Issue	● Minor	✓ Resolved

ID	Title	Category	Severity	Status
SCL-04	Rewards are not claimed and transferred in migration of strategy	Volatile Code	● Informational	✓ Resolved
SCL-05	Explicitly returning local variable	Gas Optimization	● Informational	✓ Resolved
SCS-01	Admin can change dex address	**Centralization / Privilege**	● **Informational**	ⓘ **Acknowledged**
SCS-02	Rewards are not claimed and transferred in migration of strategy	Volatile Code	● Informational	✓ Resolved
SCS-03	Usage of literal for arrays' lengths	Coding Style	● Informational	✓ Resolved
SCS-04	Inefficient storage read	Gas Optimization	● Informational	⊗ Declined
SCS-05	Explicitly returning local variable	Gas Optimization	● Informational	✓ Resolved
SES-01	Events are not emitted for state variables assignments	Volatile Code	● Informational	⊗ Declined
SES-02	Lack of validation for function parameter	Logical Issue	● Minor	✓ Resolved
STA-01	Admin can change dex address	**Centralization / Privilege**	● **Informational**	ⓘ **Acknowledged**
STA-02	Rewards are not claimed and transferred in migration of strategy	Volatile Code	● Informational	✓ Resolved
STA-03	Usage of literal for arrays' lengths	Coding Style	● Informational	✓ Resolved
STA-04	Inefficient storage read	Gas Optimization	● Informational	⊗ Declined
STA-05	Explicitly returning local variable	Gas Optimization	● Informational	✓ Resolved
STR-01	Events are not emitted for state variables assignments	Volatile Code	● Informational	⊗ Declined
STR-02	Lack of validation for function parameter	Logical Issue	● Minor	✓ Resolved
STR-03	Inefficient storage read	Gas Optimization	● Informational	⊗ Declined
TLS-01	Data location can be changed from `memory` to `calldata`	Gas Optimization	● Informational	✓ Resolved
TLS-02	Ether amount is not validated	Volatile Code	● Minor	✓ Resolved

ID	Title	Category	Severity	Status
TLS-03	Ether amount is not validated	Volatile Code	● Minor	✓ Resolved
TLS-04	Contract accepts arbitrary `ether`	Volatile Code	● Minor	⊗ Declined

SCA-01 | Admin can change dex address

Category	Severity	Location	Status
Centralization / Privilege	● **Informational**	strategies/StrategyConvexAlUsd.sol: 109	ⓘ **Acknowledged**

Description

The contract's admin has the privilege to change dex's address for each reward token.

Recommendation

No recommendations.

Alleviation

The team revisited the codebase and safe-guarded the functionality that changes dex address to be only callable through TimeLock contract. The TimeLock contract is handled by the Admin and hence the functionality to change dex address is not fully decentralized.

SCA-02 | Rewards are not claimed and transferred in migration of strategy

Category	Severity	Location	Status
Volatile Code	● Informational	strategies/StrategyConvexAlUsd.sol: 414	✓ Resolved

Description

The function on the aforementioned line migrates strategy to a new address by transferring its token balance to the new strategy address. The transferred funds does not involve the possible rewards accrued by strategy

Recommendation

We advise to revisit the `migrate` function and claim rewards before transferring the funds to new strategy address.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`. The team added boolean `claimRewardsOnMigrate`. Rewards is claimed on migrate when `claimRewardsOnMigrate` is `true`. If `false`, we will call `claimRewards` before migration. If there are significant amount of rewards to be claimed after migration, we can call `claimRewards` again, re-activate the strategy and call report.

SCA-03 | Usage of literal for arrays' lengths

Category	Severity	Location	Status
Coding Style	● Informational	strategies/StrategyConvexAlUsd.sol: 21, 28, 331, 432	✓ Resolved

Description

The aforementioned lines declare fixed length arrays and utilize integer literals to specify their lengths.

Recommendation

We advise to introduce a constant variable and utilize it to specify the lengths of fixed length arrays. This will increase the legibility of codebase.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCA-04 | Inefficient storage read

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexAlUsd.sol: 99~100	⊗ Declined

Description

The aforementioned lines read storage variable `dex[_i]` inefficiently which can optimized by storing it in a local variable and then utilizing it.

Recommendation

We advise to make use of local variables to store storage values where they are used multiple times for reducing gas costs.

Alleviation

The team did not consider the recommendation stating that the gas savings are insignificant.

SCA-05 | Explicitly returning local variable

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexAlUsd.sol: 128, 266	✓ Resolved

Description

The aforementioned lines explicitly return local variables which increases overall cost of gas.

Recommendation

Since named return variables can be declared in the signature of a function, consider refactoring to remove the local variable declaration and explicit return statement in order to reduce the overall cost of gas.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCB-01 | Admin can change dex address

Category	Severity	Location	Status
Centralization / Privilege	● **Informational**	strategies/StrategyConvexBbtc.sol: 105	ⓘ **Acknowledged**

Description

The contract's admin has the privilege to change dex's address for each reward token.

Recommendation

No recommendations.

Alleviation

The team revisited the codebase and safe-guarded the functionality that changes dex address to be only callable through TimeLock contract. The TimeLock contract is handled by the Admin and hence the functionality to change dex address is not fully decentralized.

SCB-02 | Rewards are not claimed and transferred in migration of strategy

Category	Severity	Location	Status
Volatile Code	● Informational	strategies/StrategyConvexBbtc.sol: 398	✓ Resolved

Description

The function on the aforementioned line migrates strategy to a new address by transferring its token balance to the new strategy address. The transferred funds does not involve the possible rewards accrued by strategy

Recommendation

We advise to revisit the `migrate` function and claim rewards before transferring the funds to new strategy address.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`. The team added boolean `claimRewardsOnMigrate`. Rewards is claimed on migrate when `claimRewardsOnMigrate` is `true`. If `false`, we will call `claimRewards` before migration. If there are significant amount of rewards to be claimed after migration, we can call `claimRewards` again, re-activate the strategy and call report.

SCB-03 | Usage of literal for arrays' lengths

Category	Severity	Location	Status
Coding Style	● Informational	strategies/StrategyConvexBbtc.sol: 21, 27, 314, 416	✓ Resolved

Description

The aforementioned lines declare fixed length arrays and utilize integer literals to specify their lengths.

Recommendation

We advise to introduce a constant variable and utilize it to specify the lengths of fixed length arrays. This will increase the legibility of codebase.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCB-04 | Inefficient storage read

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexBbtc.sol: 95~96	⊗ Declined

Description

The aforementioned lines read storage variable `dex[_i]` inefficiently which can optimized by storing it in a local variable and then utilizing it.

Recommendation

We advise to make use of local variables to store storage values where they are used multiple times for reducing gas costs.

Alleviation

The team did not consider the recommendation stating that the gas savings are insignificant.

SCB-05 | Explicitly returning local variable

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexBbtc.sol: 124, 249	✓ Resolved

Description

The aforementioned lines explicitly return local variables which increases overall cost of gas.

Recommendation

Since named return variables can be declared in the signature of a function, consider refactoring to remove the local variable declaration and explicit return statement in order to reduce the overall cost of gas.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCL-01 | Token approval is removed from wrong address

Category	Severity	Location	Status
Logical Issue	● Minor	strategies/StrategyCompLev.sol: 76	✓ Resolved

Description

The aforementioned line intends to remove token approval from previous dex address yet it erroneously removes token approval from the newly assigned dex address.

Recommendation

We advise to revisit the code and correctly provide the previous dex's address for the removal of token approval.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCL-02 | Admin can change dex address

Category	Severity	Location	Status
Centralization / Privilege	● **Informational**	strategies/StrategyCompLev.sol: 84	ⓘ **Acknowledged**

Description

The contract's admin has the privilege to change dex's address for each reward token.

Recommendation

No recommendations.

Alleviation

The team revisited the codebase and safe-guarded the functionality that changes dex address to be only callable through TimeLock contract. The TimeLock contract is handled by the Admin and hence the functionality to change dex address is not fully decentralized.

SCL-03 | Incorrect conditional

Category	Severity	Location	Status
Logical Issue	● Minor	strategies/StrategyCompLev.sol: 301	✓ Resolved

Description

The conditional on the aforementioned line is incorrect as if the `_targetSupply` is greater than `unleveraged` but less than `supplied` then the condition on L311 will never evaluate to `true` resulting in ineffectual call of the function.

Recommendation

We advise to revisit the conditional on `L301` such that the `_targetSupply` is greater than `supplied`.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCL-04 | Rewards are not claimed and transferred in migration of strategy

Category	Severity	Location	Status
Volatile Code	● Informational	strategies/StrategyCompLev.sol: 655	✓ Resolved

Description

The function on the aforementioned line migrates strategy to a new address by transferring its token balance to the new strategy address. The transferred funds does not involve the possible rewards accrued by strategy

Recommendation

We advise to revisit the `migrate` function and claim rewards before transferring the funds to new strategy address.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`. The team added boolean `claimRewardsOnMigrate`. Rewards is claimed on migrate when `claimRewardsOnMigrate` is `true`. If `false`, we will call `claimRewards` before migration. If there are significant amount of rewards to be claimed after migration, we can call `claimRewards` again, re-activate the strategy and call report.

SCL-05 | Explicitly returning local variable

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyCompLev.sol: 509	✓ Resolved

Description

The aforementioned lines explicitly return local variables which increases overall cost of gas.

Recommendation

Since named return variables can be declared in the signature of a function, consider refactoring to remove the local variable declaration and explicit return statement in order to reduce the overall cost of gas.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCS-01 | Admin can change dex address

Category	Severity	Location	Status
Centralization / Privilege	● **Informational**	strategies/StrategyConvexStEth.sol: 85	ⓘ **Acknowledged**

Description

The contract's admin has the privilege to change dex's address for each reward token.

Recommendation

No recommendations.

Alleviation

The team revisited the codebase and safe-guarded the functionality that changes dex address to be only callable through TimeLock contract. The TimeLock contract is handled by the Admin and hence the functionality to change dex address is not fully decentralized.

SCS-02 | Rewards are not claimed and transferred in migration of strategy

Category	Severity	Location	Status
Volatile Code	● Informational	strategies/StrategyConvexStEth.sol: 376	✓ Resolved

Description

The function on the aforementioned line migrates strategy to a new address by transferring its token balance to the new strategy address. The transferred funds does not involve the possible rewards accrued by strategy

Recommendation

We advise to revisit the `migrate` function and claim rewards before transferring the funds to new strategy address.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`. The team added boolean `claimRewardsOnMigrate`. Rewards is claimed on migrate when `claimRewardsOnMigrate` is `true`. If `false`, we will call `claimRewards` before migration. If there are significant amount of rewards to be claimed after migration, we can call `claimRewards` again, re-activate the strategy and call report.

SCS-03 | Usage of literal for arrays' lengths

Category	Severity	Location	Status
Coding Style	● Informational	strategies/StrategyConvexStEth.sol: 20, 27, 293, 392	✓ Resolved

Description

The aforementioned lines declare fixed length arrays and utilize integer literals to specify their lengths.

Recommendation

We advise to introduce a constant variable and utilize it to specify the lengths of fixed length arrays. This will increase the legibility of codebase.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SCS-04 | Inefficient storage read

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexStEth.sol: 75~76	⊗ Declined

Description

The aforementioned lines read storage variable `dex[_i]` inefficiently which can optimized by storing it in a local variable and then utilizing it.

Recommendation

We advise to make use of local variables to store storage values where they are used multiple times for reducing gas costs.

Alleviation

The team did not consider the recommendation stating that the gas savings are insignificant.

SCS-05 | Explicitly returning local variable

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexStEth.sol: 104, 230	✓ Resolved

Description

The aforementioned lines explicitly return local variables which increases overall cost of gas.

Recommendation

Since named return variables can be declared in the signature of a function, consider refactoring to remove the local variable declaration and explicit return statement in order to reduce the overall cost of gas.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

SES-01 | Events are not emitted for state variables assignments

Category	Severity	Location	Status
Volatile Code	● Informational	StrategyEth.sol: 48	ⓧ Declined

Description

The constructor on the aforementioned line assigns contract's state variables but does not emit their corresponding events.

Recommendation

We advise to emit the events corresponding to the state variables that are assigned in the body of aforementioned constructor.

Alleviation

The team did not consider our recommendation.

SES-02 | Lack of validation for function parameter

Category	Severity	Location	Status
Logical Issue	● Minor	StrategyEth.sol: 122, 132	✓ Resolved

Description

The address type parameters of the functions on aforementioned lines are used to update contract's state yet they are not validated against zero address value. If they are passed as zero address then it will result in unwanted state of the contract.

Recommendation

We advise to validate the address type function parameters of the aforementioned functions against zero address value.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

STA-01 | Admin can change dex address

Category	Severity	Location	Status
Centralization / Privilege	● Informational	strategies/StrategyConvexUsdp.sol: 107	ⓘ Acknowledged

Description

The contract's admin has the privilege to change dex's address for each reward token.

Recommendation

No recommendations.

Alleviation

The team revisited the codebase and safe-guarded the functionality that changes dex address to be only callable through TimeLock contract. The TimeLock contract is handled by the Admin and hence the functionality to change dex address is not fully decentralized.

STA-02 | Rewards are not claimed and transferred in migration of strategy

Category	Severity	Location	Status
Volatile Code	● Informational	strategies/StrategyConvexUsdp.sol: 407	✓ Resolved

Description

The function on the aforementioned line migrates strategy to a new address by transferring its token balance to the new strategy address. The transferred funds does not involve the possible rewards accrued by strategy

Recommendation

We advise to revisit the `migrate` function and claim rewards before transferring the funds to new strategy address.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`. The team added boolean `claimRewardsOnMigrate`. Rewards is claimed on migrate when `claimRewardsOnMigrate` is `true`. If `false`, we will call `claimRewards` before migration. If there are significant amount of rewards to be claimed after migration, we can call `claimRewards` again, re-activate the strategy and call report.

STA-03 | Usage of literal for arrays' lengths

Category	Severity	Location	Status
Coding Style	● Informational	strategies/StrategyConvexUsdp.sol: 21, 27, 324, 423	✓ Resolved

Description

The aforementioned lines declare fixed length arrays and utilize integer literals to specify their lengths.

Recommendation

We advise to introduce a constant variable and utilize it to specify the lengths of fixed length arrays. This will increase the legibility of codebase.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

STA-04 | Inefficient storage read

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexUsdp.sol: 97~98	⊗ Declined

Description

The aforementioned lines read storage variable `dex[_i]` inefficiently which can optimized by storing it in a local variable and then utilizing it.

Recommendation

We advise to make use of local variables to store storage values where they are used multiple times for reducing gas costs.

Alleviation

The team did not consider the recommendation stating that the gas savings are insignificant.

STA-05 | Explicitly returning local variable

Category	Severity	Location	Status
Gas Optimization	● Informational	strategies/StrategyConvexUsdp.sol: 126, 259	✓ Resolved

Description

The aforementioned lines explicitly return local variables which increases overall cost of gas.

Recommendation

Since named return variables can be declared in the signature of a function, consider refactoring to remove the local variable declaration and explicit return statement in order to reduce the overall cost of gas.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

STR-01 | Events are not emitted for state variables assignments

Category	Severity	Location	Status
Volatile Code	● Informational	Strategy.sol: 46	⊗ Declined

Description

The constructor on the aforementioned line assigns contract's state variables but does not emit their corresponding events.

Recommendation

We advise to emit the events corresponding to the state variables that are assigned in the body of aforementioned constructor.

Alleviation

The team did not consider our recommendation.

STR-02 | Lack of validation for function parameter

Category	Severity	Location	Status
Logical Issue	● Minor	Strategy.sol: 117, 127	✓ Resolved

Description

The address type parameters of the functions on aforementioned lines are used to update contract's state yet they are not validated against zero address value. If they are passed as zero address then it will result in unwanted state of the contract.

Recommendation

We advise to validate the address type function parameters of the aforementioned functions against zero address value.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

STR-03 | Inefficient storage read

Category	Severity	Location	Status
Gas Optimization	● Informational	Strategy.sol: 153~154	⊗ Declined

Description

The aforementioned lines read storage variable `fundManager` inefficiently which can optimized by storing it in a local variable and then utilizing it.

Recommendation

We advise to make use of local variables to store storage values where they are used multiple times for reducing gas costs.

Alleviation

The team decline the recommendation stating the gas savings are low.

TLS-01 | Data location can be changed from `memory` to `calldata`

Category	Severity	Location	Status
Gas Optimization	● Informational	TimeLock.sol: 76, 104	✓ Resolved

Description

The aforementioned lines specify `memory` as data location for the function parameter `data`. The `data` is received externally in `calldata` and hence the aforementioned parameters can have their data location changed to `calldata` to save gas cost associated with copying of bytes from `calldata` to `memory`.

Recommendation

We advise to change data location of the aforementioned parameters from `memory` to `calldata` to save gas cost associated with copying of parameters from `memory` to `calldata`.

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

TLS-02 | Ether amount is not validated

Category	Severity	Location	Status
Volatile Code	● Minor	TimeLock.sol: 183	✓ Resolved

Description

The function on the aforementioned line executes relayed transaction and sends `ether` along the relayed transaction yet it does not validate if the forwarding `ether` amount is received by function call or the contract has sufficient `ether` balance.

Recommendation

We advise to introduce a check ensuring that either the function call received the forwarding `ether` or the contract has sufficient balance to successfully execute the relayed call.

```
require(
    msg.value == value
    || value <= address(this).balance,
    "not enough ether balance"
);
```

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

TLS-03 | Ether amount is not validated

Category	Severity	Location	Status
Volatile Code	● Minor	TimeLock.sol: 196	✓ Resolved

Description

The function on the aforementioned line executes relayed transactions and sends `ether` along the relayed transactions yet it does not validate if the forwarding `ether` amount is received by function call or the contract has sufficient `ether` balance.

Recommendation

We advise to introduce a check ensuring that either the function call received the forwarding `ether` or the contract has sufficient balance to successfully execute the relayed call.

```
uint256 requiredEtherBalance;
for (uint i = 0; i < targets.length; i++) {
    requireEtherBalance += values[i];
}

require(
    msg.value == requiredEtherBalance
    || requiredEtherBalance <= address(this).balance,
    "not enough ether balance"
);
```

Alleviation

Alleviations are applied as of commit hash `0cdc6074ac49797b3d5a30d5243caefd29fb0563`.

TLS-04 | Contract accepts arbitrary `ether`

Category	Severity	Location	Status
Volatile Code	● Minor	TimeLock.sol: 38	⊗ Declined

Description

The `receive` function on the aforementioned line allows contract to accept arbitrary `ether`.

Recommendation

We advise to introduce a check ensuring that only a whitelisted address is able to sent plain `ether` to avoid any address from mistakenly sending the `ether`.

Alleviation

The team did not consider the recommendation stating "Accidentally sent ETH can be sent back by time lock (queue + execute)".

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we’re able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

CERTIK