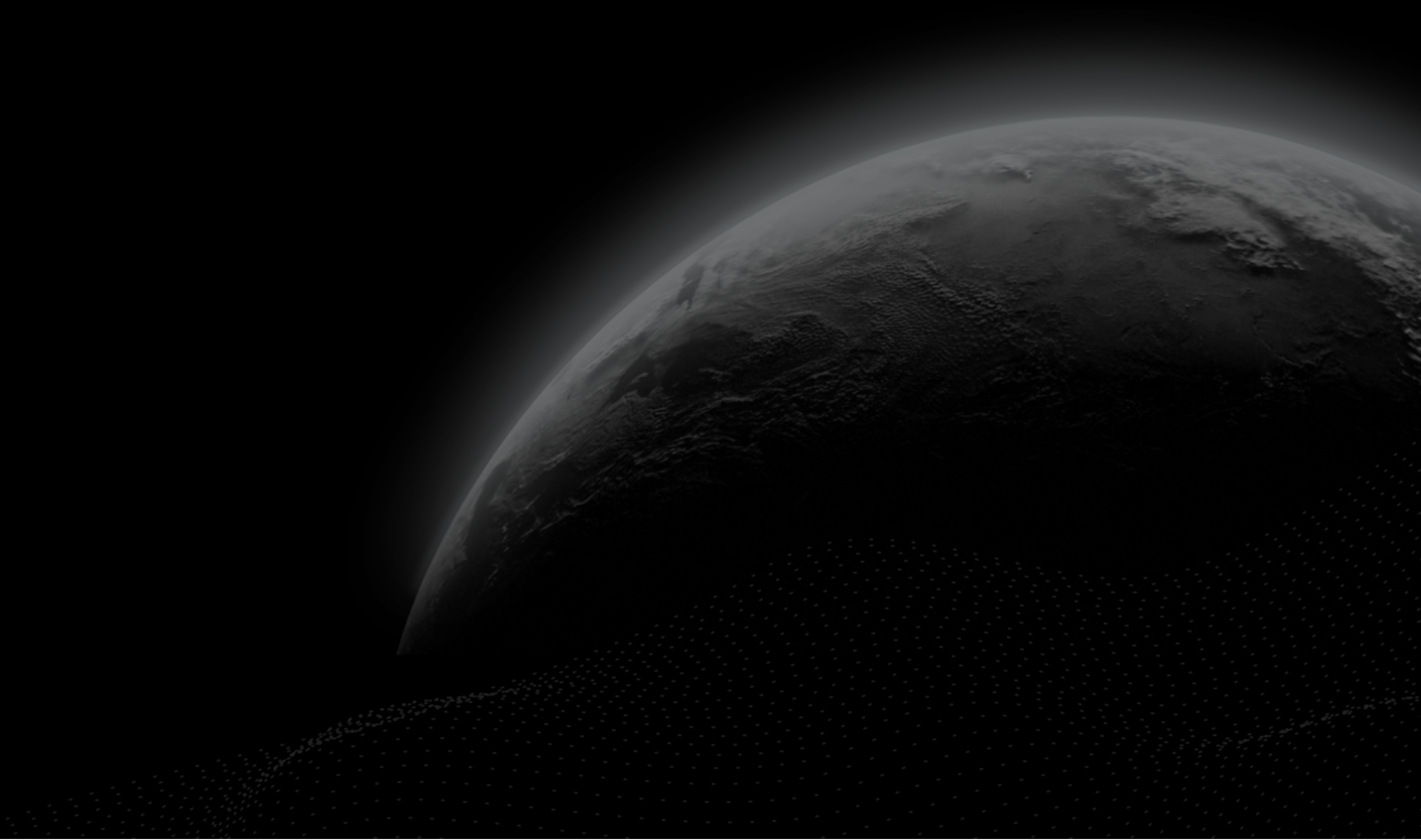
CERTIK

Security Assessment

Venus - Prime and Oracle Changes

CertiK Assessed on Dec 19th, 2023



CertiK Assessed on Dec 19th, 2023

Venus - Prime and Oracle Changes

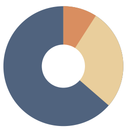
The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Binance Smart Chain (BSC)	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 12/19/2023	N/A

CODEBASE	COMMITS
PR-128: https://github.com/VenusProtocol/oracle/pull/128	base-PR-128: 5e2dcbf33e92fe0865134653f87779f06f563083
PR-142: https://github.com/VenusProtocol/oracle/pull/142	base-PR-142: 16288e9d642f9fd6ce226cd9aec25b6e6c577315
PR-327: https://github.com/VenusProtocol/isolated-pools/pull/327	base-PR-327: 6b600e7caec67c34476da8cb62ee17c0b052f67f
View All in Codebase Page	View All in Codebase Page

Vulnerability Summary



11	7	1	1	2	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

🟥 0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧 1	Major	1 Mitigated	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨 0	Medium		Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨 3	Minor	1 Resolved, 1 Partially Resolved, 1 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦 7	Informational	6 Resolved, 1 Acknowledged	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | VENUS - PRIME AND ORACLE CHANGES

CODEBASE | VENUS - PRIME AND ORACLE CHANGES

Repository

PR-128: https://github.com/VenusProtocol/oracle/pull/128

PR-142: https://github.com/VenusProtocol/oracle/pull/142

PR-327: https://github.com/VenusProtocol/isolated-pools/pull/327

PR-407: https://github.com/VenusProtocol/venus-protocol/pull/407 \

Commit

base-PR-128: 5e2dcbf33e92fe0865134653f87779f06f563083

base-PR-142: 16288e9d642f9fd6ce226cd9aec25b6e6c577315

base-PR-327: 6b600e7caec67c34476da8cb62ee17c0b052f67f

base-PR-407 : 0a51f8461c4546fb5cb90d9672cafec90cc59714

AUDIT SCOPE | VENUS - PRIME AND ORACLE CHANGES

14 files audited ● 5 files with Acknowledged findings ● 2 files with Partially Resolved findings ● 1 file with Mitigated findings ● 2 files with Resolved findings ● 4 files without findings

ID	Repo	File	SHA256 Checksum
● IPP	VenusProtocol/venus-protocol	Tokens/Prime/IPrime.sol	5aebceb231a957cb6c4250b12f78ee48c2914bb657d7e70b3c5f9d5a4ae8510c
● IPI	VenusProtocol/venus-protocol	Tokens/Prime/Interfaces/IPrime.sol	fc34ef11c4116035f549723c6ea8046c07e351e3e331853c8ae1a6d30b735bef
● CVP	VenusProtocol/isolated-pools	Comptroller.sol	816fc7060b9897813a15d35c97f73e5a7f87b4798dfb23390754140261447409
● VTV	VenusProtocol/isolated-pools	VToken.sol	d9d7de0605258188d83d7756c0097a3c4c973750f6896ad8a438604d83a72d87
● BOV	VenusProtocol/oracle	oracles/BinanceOracle.sol	79860f916467b41f4956f4dfe6d5accca0099a8ee96025ca4382a1e33b85e809
● PPT	VenusProtocol/venus-protocol	Tokens/Prime/Prime.sol	0d5265e565cebc0efea63a53b5a33020d265b50e327e9d454ee758542556414f
● VAI	VenusProtocol/venus-protocol	Tokens/VAI/VAIController.sol	f78f34320d146cdfd51bd5b8d01a40dcd5b8cf4dd8d3970aedb2ae98f69a2cd8
● TMU	VenusProtocol/venus-protocol	Utils/TimeManager.sol	51206f8919ad43364981ff6039bc403ed31f77ae1186ac5b1a5cfb0604fe76ed
● PLP	VenusProtocol/venus-protocol	Tokens/Prime/PrimeLiquidityProvider.sol	55c9b66d4c23af5c1c66a72d22f3d275c06c4e5b9f04de647c1ebca36d82ce7f
● VAC	VenusProtocol/venus-protocol	Tokens/VAI/VAIControllerStorage.sol	4a21e64ad56850a0ee82e6cac249dcd8471192963ddac8dc7a274f20769be700
● PSP	VenusProtocol/venus-protocol	Tokens/Prime/PrimeStorage.sol	376182a4a66b5e24999473496ca6d5580709b981b5715ef8c50b6df20660eeec
● CSV	VenusProtocol/isolated-pools	ComptrollerStorage.sol	553043abda7a286ddd628f253af21c0568a9b28d4e854b902112903a10cdaac5
● ACO	VenusProtocol/oracle	oracles/ArbiChainlinkOracle.sol	5992200fd387ab7a11b2e2d43e2e020932c11cfc6ff6906b22413256670016a2

ID	Repo	File	SHA256 Checksum
● COV	VenusProtocol/oracle	oracles/ChainlinkOracle.sol	dcd4cf37706547be5424b02d040b3bbb4a142962ba7e4a31c9893686ffa13c1d

APPROACH & METHODS | VENUS - PRIME AND ORACLE CHANGES

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

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

SUMMARY | VENUS - PRIME AND ORACLE CHANGES

This audit concerns the changes made in files outlined in:

- *Venus Prime* PR-407, commit 0a51f8461c4546fb5cb90d9672cafec90cc59714 compared against the last audited commit 4eac8359e3364df5898cb4b85b17f6f4c1f71b65.
- *Isolated Pools* PR-327, commit 6b600e7caec67c34476da8cb62ee17c0b052f67f
- *Oracle* PR-142, commit 16288e9d642f9fd6ce226cd9aec25b6e6c577315
- *Oracle* PR-128, commit 5e2dcbf33e92fe0865134653f87779f06f563083

Note that any centralization risks present in the existing codebase before these PRs were not considered in this audit and only those added in these PRs are addressed in the audit. We recommend all users to carefully review the centralization risks, much of which can be found in our previous audits *Venus - Prime*, *Venus - Oracle*, and *Venus - Isolated Pools* which can be found here: https://skynet.certik.com/projects/venus.

DEPENDENCIES | VENUS - PRIME AND ORACLE CHANGES

Third Party Dependencies

The protocol is serving as the underlying entity to interact with third party protocols. The third parties that the contracts interact with are:

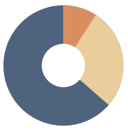
- ERC20 Tokens
- Oracles

The scope of the audit treats third party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets. Moreover, updates to the state of a project contract that are dependent on the read of the state of external third party contracts may make the project vulnerable to read-only reentrancy. In addition, upgrades of third parties can possibly create severe impacts, such as increasing fees of third parties, migrating to new LP pools, etc.

Recommendations

We recommend constantly monitoring the third parties involved to mitigate any side effects that may occur when unexpected changes are introduced, as well as vetting any third party contracts used to ensure no external calls can be made before updates to its state.

FINDINGS | VENUS - PRIME AND ORACLE CHANGES



11	0	1	0	3	7
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Venus - Prime and Oracle Changes . Through this audit, we have uncovered 11 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
VPU-03	**Centralization Related Risks**	**Centralization**	**Major**	● **Mitigated**
BOV-01	Either `sidRegistryAddress` Or `feedRegistryAddress` Should Be Set Upon Initialization	Logical Issue	Minor	● Acknowledged
PPT-01	Wrong String For Access Allowed Check	Logical Issue	Minor	● Resolved
VPH-01	Missing Input Validation	Logical Issue	Minor	● Partially Resolved
BOV-02	Emit Event Pattern Inconsistency	Inconsistency	Informational	● Resolved
IPI-01	Not All External Facing Functions Are Represented In `IPrime`	Inconsistency	Informational	● Resolved
PLP-01	Language Is Not Consistent	Inconsistency	Informational	● Resolved
PTV-01	Duplicate File Name	Inconsistency	Informational	● Acknowledged
VAI-01	Naming Convention Inconsistency	Inconsistency	Informational	● Resolved
VAT-01	Specific Imports Not Consistently Used	Inconsistency	Informational	● Resolved

ID	Title	Category	Severity	Status
VPH-02	Typos And Inconsistencies	Coding Style	Informational	● Resolved

VPU-03 | CENTRALIZATION RELATED RISKS

Category	Severity	Location	Status
Centralization	● **Major**	**oracles/BinanceOracle.sol (base-PR-142): 94~105; Tokens/VAI/VAIController.sol (base-PR-407): 401~404**	● **Mitigated**

Description

The centralization risks indicated here are only related to those within the scope of the delta audit. CertiK has audited much of the codebase before and their relevant centralization risks can be found in our audit reports here: https://skynet.certik.com/projects/venus. For those contracts that have not been audited by CertiK, we recommend reviewing the contracts and carefully considering the centralization risks present.

BinanceOracle

In the contract `BinanceOracle` the role `_owner` has authority over the function `setFeedRegistryAddress()` . Any compromise to the `_owner` may allow the hacker to change the feed registry address to a malicious contract and return incorrect prices. In the worst case scenario, this can be used to steal all borrowable funds from the protocol.

VAIController

In the contract `VAIController` the role `admin` has authority over the function `_setPrimeToken()` . Any compromise to the `admin` may allow the hacker to change the prime token address preventing those that hold true prime tokens from minting VAI and allowing them to mint VAI when not holding a prime token.

Isolated Pools Comptroller

In the contract `Comptroller` , the role `_owner` has authority over the function `setPrimeToken()` . Any compromise to the `_owner` may allow the hacker to change the prime token address, preventing the timely update of scores due to changes in any user's market interactions.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

`[Venus, 12/15/2023]` : The BinanceOracle is already deployed to BNB chain 1, and the owner is the Normal Timelock 2. In new BinanceOracle contracts, deployed to new networks, the owner will be a Normal Timelock contract.

The VAIController is already deployed to BNB chain 3, the admin is the Normal Timelock 2. If we would deploy new VAIController contracts to other networks, the owner would be a Normal Timelock contract.

There are several Comptroller contracts deployed to BNB mainnet [4], and the owner is the Normal Timelock in every case. In new Comptroller contracts, deployed to new networks, the owner will be also a Normal Timelock contract.

[4]:

https://bscscan.com/address/0x94c1495cD4c557f1560Cbd68EAB0d197e6291571

https://bscscan.com/address/0x3344417c9360b963ca93A4e8305361AEde340Ab9

https://bscscan.com/address/0x1b43ea8622e76627B81665B1eCeBB4867566B963

https://bscscan.com/address/0xd933909A4a2b7A4638903028f44D1d38ce27c352

https://bscscan.com/address/0x23b4404E4E5eC5FF5a6FFb70B7d14E3FabF237B0

BOV-01 | EITHER `sidRegistryAddress` OR `feedRegistryAddress` SHOULD BE SET UPON INITIALIZATION

Category	Severity	Location	Status
Logical Issue	● Minor	oracles/BinanceOracle.sol (base-PR-142): 57~65	● Acknowledged

Description

If the current chain supports space ID, then `sidRegistryAddress` should be set upon initialization. Otherwise, the `feedRegistryAddress` should be set upon initialization or the oracle will revert when calling `latestRoundDataByName()` on the zero address.

Recommendation

We recommend adding an input `_feedRegistryAddress` to set the `feedRegistryAddress` to and if the input `_sidRegistryAddress` is `address(0)` to ensure that `_feedRegistryAddress` is not `address(0)` .

Alleviation

`[Venus, 12/15/2023]` : "Issue acknowledged. I won't make any changes for the current version.

It would require maintaining two codebases because of any change in initialier force in using "reinitializer(2)" on BNB chain. That's the reason why we avoided this new variable in the "initialize" function.

Moreover, we are setting the FeedRegistryAddress, if needed, in the deployment script: https://github.com/VenusProtocol/oracle/blob/develop/deploy/1-deploy-oracles.ts#L158"

PPT-01 | WRONG STRING FOR ACCESS ALLOWED CHECK

Category	Severity	Location	Status
Logical Issue	● Minor	Tokens/Prime/Prime.sol (base-PR-407): 393	● Resolved

Description

The input `comptroller` was added to the function `addMarket()` , which is controlled by the ACM. However, the input string for `_checkAccessAllowed()` does not include the new address parameter.

Recommendation

We recommend adding another address parameter to the check access string.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit d493a3dc11c8ba42c6c013b054fdbeb6b0bd6ea0.

VPH-01 | MISSING INPUT VALIDATION

Category	Severity	Location	Status
Logical Issue	● Minor	Tokens/Prime/Prime.sol (base-PR-407): 160~161, 388, 398; Utils/TimeManager.sol (base-PR-407): 27~29	● Partially Resolved

Description

Prime

- In function `addMarket()` , there is no check that the input `comptroller` represents a pool supported by the protocol. In particular if it is an isolated pool it should be registered in the pool registry and if not then it should be the main pool.
- In the constructor, there is no check that the addresses used to set `WRAPPED_NATIVE_TOKEN` and `NATIVE_MARKET` are nonzero.

TimeManager.sol

- The logic of the constructor makes the check that if `timeBased_` is false, then `blocksPerYear` is nonzero in that case. In order to ensure the intended set up in all cases, a check should also be included to ensure that if `timeBased_` is true, then `blocksPerYear` is zero in that case.

Recommendation

We recommend including the checks outlined above.

Alleviation

`[Venus, 12/15/2023]` : "WRAPPED_NATIVE_TOKEN, NATIVE_MARKET and corePoolComptroller will be zero address in other chains. poolRegistry will be zero address in binance chain"

`[CertiK, 12/18/2023]` : The client made changes partially resolving the finding in the following commits

- e47ef15d223d64a4b6d18092e501af0ee85d69d9
- 0701f27ff4cbd35d4de5807f20f34418b9aeccd0
- 5eb9df469b18b1626d8da89fa3420f4908b3ab1e
- 79471425a98e90c048240121122c6b877fbb2fce

BOV-02 | EMIT EVENT PATTERN INCONSISTENCY

Category	Severity	Location	Status
Inconsistency	● Informational	oracles/BinanceOracle.sol (base-PR-142): 102~104	● Resolved

Description

Throughout the codebase when emitting events for addresses changing, the convention is to first emit the event with the old address being the current state of the variable and the new address being the input, and then afterwards setting the variable to the input.

Recommendation

We recommend using the same convention to be consistent.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit 59c86ae6a1320d7caa7e6e814fe80930ec25ea04.

IPI-01 | NOT ALL EXTERNAL FACING FUNCTIONS ARE REPRESENTED IN `IPrime`

Category	Severity	Location	Status
Inconsistency	● Informational	Tokens/Prime/Interfaces/IPrime.sol (base-PR-407): 10	● Resolved

Description

The interface for `IPrime` does not include all external facing functions.

Recommendation

We recommend including all external facing functions to the interface.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit e583d9c179dc0b766cd64dd4f269cbfce1ca0899.

PLP-01 | LANGUAGE IS NOT CONSISTENT

Category	Severity	Location	Status
Inconsistency	● Informational	Tokens/Prime/PrimeLiquidityProvider.sol (base-PR-407): 31, 37~38, 55, 187, 213, 265, 309, 311, 335, 336, 366, 367, 374, 382, 402, 413, 428	● Resolved

Description

The `PrimeLiquidityProvider` contract now inherits `TimeManager` to allow the use of block numbers or timestamps. As such the language referring to block numbers should now reflect that it is a block number or possibly a timestamp.

Recommendation

We recommend adjusting the naming to indicate that it is a block or timestamp.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit c592220b8fcc958e7dc333b71f2a4c0586b52e41.

PTV-01 | DUPLICATE FILE NAME

Category	Severity	Location	Status
Inconsistency	● Informational	Tokens/Prime/IPrime.sol (base-PR-407): 10; Tokens/Prime/Interfaces/IPrime.sol (base-PR-407): 10	● Acknowledged

Description

There are two interfaces named `IPrime` . One interface has the majority of the external facing functions of `Prime` , however, the other interface only has the functions necessary for the comptroller to call during the verify hooks.

Recommendation

We recommend renaming one of the files to distinguish them easily.

Alleviation

`[Venus, 12/15/2023]` : "Issue acknowledged. I won't make any changes for the current version.

Due to different solidity versions compatibility we have two different interface files"

VAI-01 | NAMING CONVENTION INCONSISTENCY

Category	Severity	Location	Status
Inconsistency	● Informational	Tokens/VAI/VAIController.sol (base-PR-407): 401	● Resolved

Description

Other external functions that have the `onlyAdmin` modifier do not include a leading underscore.

Recommendation

We recommend removing the leading underscore for consistency.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit 2cd49f19a056fc8ffd67bd5830625d8fc9d5e683.

VAT-01 | SPECIFIC IMPORTS NOT CONSISTENTLY USED

Category	Severity	Location	Status
Inconsistency	● Informational	Tokens/VAI/VAIController.sol (base-PR-407): 3~11; Tokens/VAI/VAIControllerStorage.sol (base-PR-407): 3	● Resolved

Description

Many of the added files use specific imports, however, some import the entire file.

Recommendation

We recommend using specific imports to clarify what is used and remain consistent.

Alleviation

`[CertiK, 12/15/2023]` : The client made changes resolving the finding in commits

- f668c8153144396dfaefc5330ff9d2cef3d779df
- b143947d4a93edc71d4c85ad7404db8a339b9ac6

VPH-02 | TYPOS AND INCONSISTENCIES

Category	Severity	Location	Status
Coding Style	● Informational	Tokens/Prime/Prime.sol (base-PR-407): 137; Utils/TimeManager.sol (base-PR-407): 38~39	● Resolved

Description

TimeManager.sol

- Function `getBlockNumberOrTimestamp()` includes the following comment above its declaration: "This exists mainly for inheriting test contracts to stub this result." This contract and this function specifically will be used in production-level contracts.

Prime.sol

- The comment above error `InvalidTimestamp` misspells the word "invalid" as "invalud."

Recommendation

We recommend correcting the typos and inconsistencies outlined above.

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commits

- 2a8d05448b2c20f2b6d0d0f2c97b3749f1deacb5
- ba0ab11b923d188f7e43405741bffbb903d828f4

OPTIMIZATIONS | VENUS - PRIME AND ORACLE CHANGES

ID	Title	Category	Severity	Status
CVP-01	Unnecessary Check In `redeemVerify()`	Code Optimization, Design Issue	Optimization	● Resolved
VPB-01	Unused Parameters	Code Optimization	Optimization	● Acknowledged

CVP-01 | UNNECESSARY CHECK IN `redeemVerify()`

Category	Severity	Location	Status
Code Optimization, Design Issue	● Optimization	Comptroller.sol (base-PR-327): 359~360	● Resolved

Description

The following check is made in the newly added `redeemVerify()` logic of the `Comptroller` contract for Isolated Pools:

```
        if (redeemAmount == 0 && redeemTokens == 0) revert NoRedeemTokensOrAmount();
```

This check requires one of `redeemAmount` or `redeemTokens` is nonzero. The check is unnecessary because the logic within the function `_redeemFresh()` of the Isolated Pools VToken contract ensures that either both values `redeemAmount` and `redeemTokens` are positive, or else the function call reverts (justification for this claim is outlined below).

In particular, the check in `redeemVerify()` alone would not prevent the case where `redeemTokens` is 0 while `redeemAmount` is positive, which is the case prevented in the core pool's `redeemVerify()` function:

```
        require(redeemTokens != 0 || redeemAmount == 0, "redeemTokens zero");
```

The justification below also shows why the check done in the Core pool `redeemVerify()` function is unneeded for the Isolated Pool `redeemVerify()` function - namely, that there is never a case where `redeemTokens` is 0 while `redeemAmount` is positive.

Justification

The `_redeemFresh()` function of the Isolated Pools `VToken` contract ensures that `redeemAmount` and `redeemTokens` are either both positive, or the function reverts.

The input parameters `redeemTokensIn` and `redeemAmountIn` provided by the user are required to be values where at least one is zero.

Case 1 `redeemTokensIn` is 0 and `redeemAmountIn` is positive.

Then `redeemTokens = div_(redeemAmountIn, exchangeRate)`.

If `exchangeRate > redeemAmountIn`, then `redeemTokens` is 0. If that is true, then `redeemAmount = mul_ScalarTruncate(exchangeRate, redeemTokens); = 0` where the `redeemTokens` value of 0 is used, so the check that

```
if (redeemAmount == 0) {
            revert("redeemAmount is zero");
        }
```

will cause a revert.

If `redeemTokens` is set to a positive value, then either `redeemAmount` is positive or 0. If `redeemAmount` is 0, then a revert will occur for the same reason above, so necessarily, both values will be positive if used within the `redeemVerify()` function call at the end.

Case 2 `redeemAmountIn` is 0 and `redeemTokensIn` is positive.

Then `redeemTokens = redeemTokensIn > 0` and `redeemAmount = mul_ScalarTruncate(exchangeRate, redeemTokens)`.

If `redeemAmount` is zero, then the check outlined above will cause a revert so that both values will be positive if used within the `redeemVerify()` function call at the end.

Case 3 both `redeemTokensIn` and `redeemAmountIn` are 0.

Then `redeemTokens = div_(redeemAmountIn, exchangeRate) = 0` and consequently `redeemAmount` will be 0 (like in case 1), causing a revert.

Recommendation

We recommend removing the unnecessary check in the `redeemVerify()` logic of the Isolated Pools `Comptroller` contract

Alleviation

`[CertiK, 12/18/2023]` : The client made changes resolving the finding in commit 88de67386e849e5af26ea5e2a380dc90b007a2ad.

VPB-01 | UNUSED PARAMETERS

Category	Severity	Location	Status
Code Optimization	● Optimization	Comptroller.sol (base-PR-327): 314~315, 315~316, 318, 318, 369~370, 371~372, 375~376, 377~378, 378~379, 388~389, 391~392, 392~393, 411~412, 414~415, 434~435, 437, 519~520, 522; VToken.sol (base-PR-327): 851, 851, 987, 1040, 1040, 1040, 1156~1157, 1159~1160, 1160~1161, 1220, 1220, 1433	● Acknowledged

Description

The verify hooks are being added into the `Comptroller` logic during the upgrade corresponding to this audit. It is unnecessary to include parameters which are unused in the verify logic. They can be added back in a future upgrade if they are ever needed.

Recommendation

We recommend removing the passing of unused parameters between the `Comptroller` and `VToken` contracts.

Alleviation

`[Venus, 12/15/2023]` : "Issue acknowledged. I won't make any changes for the current version. We prefer to keep these parameters defined now, we might use them in the future and we think being explicit now could avoid errors later."

APPENDIX | VENUS - PRIME AND ORACLE CHANGES

Finding Categories

Categories	Description
Coding Style	Coding Style findings may not affect code behavior, but indicate areas where coding practices can be improved to make the code more understandable and maintainable.
Inconsistency	Inconsistency findings refer to different parts of code that are not consistent or code that does not behave according to its specification.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.
Design Issue	Design Issue findings indicate general issues at the design level beyond program logic that are not covered by other finding categories.

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 | CERTIK

UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED “AS IS” AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | **Securing** the **Web3** World

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.