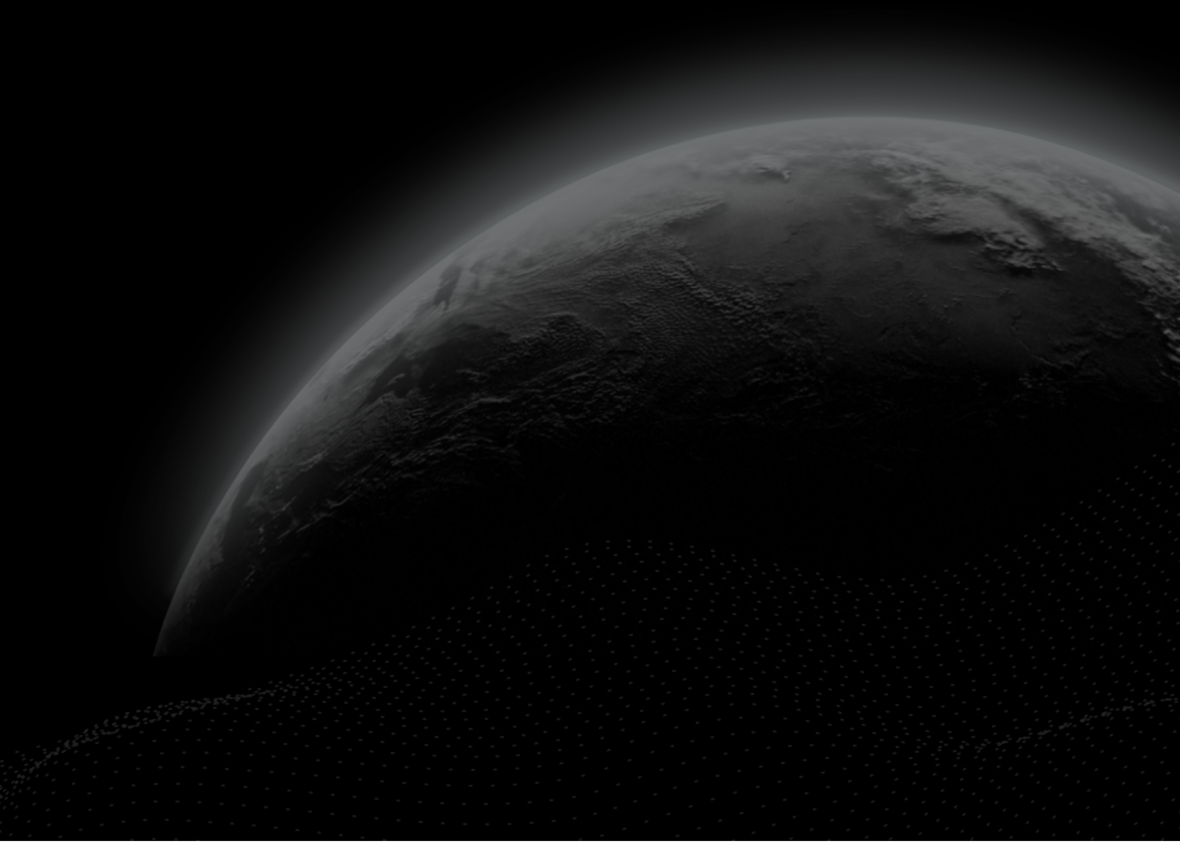
CERTIK

Security Assessment

# Venus - Time based contracts & SeizeVenus

CertiK Assessed on Jan 17th, 2024



CertiK Assessed on Jan 17th, 2024

# Venus - Time based contracts & SeizeVenus

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

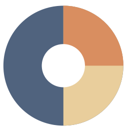
## Executive Summary

| TYPES | ECOSYSTEM | METHODS |
|---|---|---|
| DeFi | Binance Smart Chain (BSC) | Manual Review, Static Analysis |
| **LANGUAGE** | **TIMELINE** | **KEY COMPONENTS** |
| Solidity | Delivered on 01/17/2024 | N/A |

CODEBASE
https://github.com/VenusProtocol/venus-protocol
https://github.com/VenusProtocol/isolated-pools
View All in Codebase Page

COMMITS
PR324-Base: 3063e64c7757ea15a9832b738f36710a78a0627d
PR337-Base: 63dee7f678d57a2e4755312d63568bb3fb43bbd7
PR418-Base: e6fcaaef0b3928f801c4372d2ef229ef8ac8c550
View All in Codebase Page

## Vulnerability Summary



| 8 | 7 | 0 | 0 | 1 | 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. |
| 🟧 2 | Major | 1 Resolved, 1 Acknowledged | 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. |
| 🟨 2 | Minor | 2 Resolved | 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. |
| 🟦 4 | Informational | 4 Resolved | 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 - TIME BASED CONTRACTS & SEIZEVENUS

# CODEBASE | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

## Repository

https://github.com/VenusProtocol/venus-protocol

https://github.com/VenusProtocol/isolated-pools

## Commit

PR324-Base: 3063e64c7757ea15a9832b738f36710a78a0627d

PR337-Base: 63dee7f678d57a2e4755312d63568bb3fb43bbd7

PR418-Base: e6fcaaef0b3928f801c4372d2ef229ef8ac8c550

PR414-Base: 53e37eb614ad9e23a74f1d159f28c5e311175561

PR417-Base: c5ff28206f543a2e718fa5974fb0d71963324043

PR410-Base: 426462dba950bca5f2d87104c9947e8c6179d057

PR324-Update1: c749e0d6d301357876806274d244e06da0cb8108

PR414-Update1: 9cfeba718e68aa7294c9895c51037f9e9b81e450

PR417-Update1: 0c7e1f8ea0e3453c530dbcafa5be5849d62748ba

PR410-Update1: b344f3db895302c499cabba33dcd9541548d06b5

PR324-Update2: e16e4c7cf0283295602c38847a24f56a9d7900d1

PR417-Update2: c86e591f7a037437b4dd6a024366b8d9535a6acf

# AUDIT SCOPE | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

18 files audited ● 4 files with Acknowledged findings ● 7 files with Resolved findings ● 7 files without findings

| ID | Repo | File | SHA256 Checksum |
|---|---|---|---|
| ● RDR | VenusProtocol/isolated-pools | Rewards/RewardsDistributor.sol | c107a7789ad2a48eea3e15e6b904ed3f4ab01904bdeb97fe51988a93b343fa52 |
| ● JRM | VenusProtocol/isolated-pools | JumpRateModelV2.sol | 42e2ef0345678d349c2ff1e40cdf601529ee95b1cc408b0d393752d4e8279a29 |
| ● XVS | VenusProtocol/venus-protocol | XVSVault.sol | 16112c18a0c3cf6fe6101cb57a85d7171883413fc6768937beacc5a2d472d5c0 |
| ● SFD | VenusProtocol/venus-protocol | Diamond/facets/SetterFacet.sol | d5df070f0c1519ab76df7ee54b39e9a1f0e637a66f84850c790cc1c83ae7488a |
| ● WPI | VenusProtocol/isolated-pools | WhitePaperInterestRateModel.sol | d92339969e1c632597b0ee5924beae5e1fb7feda2efb5f0f7de204da707d9778 |
| ● VTV | VenusProtocol/isolated-pools | VToken.sol | 9957cbb5c1b80b2601cd6d1172f2f27385f85ddc884dc208be75cb38aa03a415 |
| ● VTI | VenusProtocol/isolated-pools | VTokenInterfaces.sol | 6935f492b98e7d5caad3bbbee45977f289bfd4a362e3e17ba7a26e33ca25482a |
| ● SSV | VenusProtocol/isolated-pools | Shortfall/Shortfall.sol | 026b5a59fba3015f187aa954a32bd2b1a54e6e290ae7a0f798519765df5d9100 |
| ● PLL | VenusProtocol/isolated-pools | Lens/PoolLens.sol | 78ce28ed35a050eecf8ae63783d0101bc5f1b691c02256f5dcc622aa17017e64 |
| ● FBD | VenusProtocol/venus-protocol | Diamond/facets/FacetBase.sol | 6dbbd65d52ab04481be27090a4432671a1baacc8e0f27053ad25d9d603fd6dd3 |
| ● RFC | VenusProtocol/venus-protocol | Diamond/facets/RewardFacet.sol | f248a05cbafaea9f6bfdd511c13340b1b410ab1ca0bbd5d7d49e99650c5eef5c |
| ● VPB | VenusProtocol/isolated-pools | lib/constants.sol | da077999bb442480eddb09a25a7c35e6af2484ce2f2e6749e6707fe77f4800e8 |
| ● VTP | VenusProtocol/isolated-pools | VToken.sol | 4e583c1c2446fc54e0bcdaa3190852cf65888a3ee23f1682a2a87f6f41b9c60f |

| ID | Repo | File | SHA256 Checksum |
|---|---|---|---|
| TMV | VenusProtocol/venus-protocol | TimeManagerV5.sol | cf1a21e94fd7cb9ef5a02b24f866bf4acb7cc8859626add6b4f75a090396f682 |
| XVV | VenusProtocol/venus-protocol | XVSVaultStorage.sol | 54344f072b2c340d1e0b2c21a22bfc85508660736d4b7cab096a9c645b66a65d |
| VTT | VenusProtocol/venus-protocol | VToken.sol | e06897c5229b20459d5c8b342cd19f9dbcc14be9b1d4d72a6a2747ecffbfdc24 |
| RFD | VenusProtocol/venus-protocol | RewardFacet.sol | 28c968fd4bdd8699009886ec570ae164f888172ceb1422d36905bc4ed0159dc2 |
| CSC | VenusProtocol/venus-protocol | ComptrollerStorage.sol | f2863250b064cbde11e8775b6ba72dc956e2f1dffee3d1d65671a6ad374b906c |

# APPROACH & METHODS | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

This report has been prepared for Venus to discover issues and vulnerabilities in the source code of the Venus - Time based contracts & SeizeVenus 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 - TIME BASED CONTRACTS & SEIZEVENUS

This audit concerns the changes made in files outlined in:

- *Isolated Pools* PR-324, commit 3063e64c7757ea15a9832b738f36710a78a0627d
- *Isolated Pools* PR-337, commit 63dee7f678d57a2e4755312d63568bb3fb43bbd7
- *Venus Protocol* PR-418, commit e6fcaaef0b3928f801c4372d2ef229ef8ac8c550
- *Venus Protocol* PR-414, commit 53e37eb614ad9e23a74f1d159f28c5e311175561
- *Venus Protocol* PR-417, commit c5ff28206f543a2e718fa5974fb0d71963324043
- *Venus Protocol* PR-410, commit 426462dba950bca5f2d87104c9947e8c6179d057

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 which can be found here: https://skynet.certik.com/projects/venus.

# DEPENDENCIES | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

## 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:

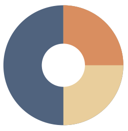
- ERC20 Tokens
- Oracles

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 - TIME BASED CONTRACTS & SEIZEVENUS



| 8 | 0 | 2 | 0 | 2 | 4 |
|---|---|---|---|---|---|
| Total Findings | Critical | Major | Medium | Minor | Informational |

This report has been prepared to discover issues and vulnerabilities for Venus - Time based contracts & SeizeVenus. Through this audit, we have uncovered 8 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 |
|---|---|---|---|---|
| VPB-02 | Potential Storage Collision | Coding Issue | Major | ● Resolved |
| **VPU-02** | **Centralization Related Risks** | **Centralization** | **Major** | ● **Acknowledged** |
| SSV-01 | Block Number-Based Variables Should Be Made Immutable | Design Issue | Minor | ● Resolved |
| VPB-03 | Maximum Should Depend On Chain And Whether The Contracts Are Time Or Block-Based | Logical Issue | Minor | ● Resolved |
| RFD-04 | Missing Or Incomplete NatSpec | Inconsistency | Informational | ● Resolved |
| VPB-01 | Value Naming Should Be Updated To Reflect Possible Time-Based Case | Coding Style | Informational | ● Resolved |
| VPH-01 | Typos And Inconsistencies | Inconsistency | Informational | ● Resolved |
| VTT-01 | Missing Return Statement In `setReduceReservesBlockDelta()` And `setProtocolShareReserve()` | Coding Issue | Informational | ● Resolved |

## VPB-02 | POTENTIAL STORAGE COLLISION

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Issue | ● Major | JumpRateModelV2.sol (PR324-Base): 15~16; Lens/PoolLens.sol (PR324-Base): 29~30; Rewards/RewardsDistributor.sol (PR324-Base): 31~32; Shortfall/Shortfall.sol (PR324-Base): 31~32; VToken.sol (PR324-Base): 45~46; WhitePaperInterestRateModel.sol (PR324-Base): 13~14 | ● Resolved |

### Description

In PR 324, contract `TimeManagerV8` is being imported for use within each of the in-scope contracts. Given the contract's order of inheritance within these contracts, and that the contract contains a `__gap` storage variable, it will cause storage collisions if it is used to upgrade any existing deployments.

### Recommendation

We recommend adjusting the storage so that it is compatible with the previous deployments and will not cause any storage collisions.

### Alleviation

`[CertiK, 01/12/2024]` : The client resolved the finding in commits

- 59b141362bb92ac0a7f2aa528107fd62f327abf0;
- d6311f25cf84ac1c82a143109e45ad9fea1d4dc3

`[Venus, 01/16/2024]` : "The contract JumpRateModelV2 is not deployed via proxy, which means it is non-upgradable. So, no storage collision will occur."

## VPU-02 | CENTRALIZATION RELATED RISKS

| Category | Severity | Location | Status |
|---|---|---|---|
| **Centralization** | ● **Major** | **JumpRateModelV2.sol (PR324-Base): 88~95; Rewards/RewardsDistributor.sol (PR324-Base): 299~304; Diamond/facets/SetterFacet.sol (PR410-Base): 584~585, 596~597; RewardFacet.sol (PR417-Base): 145~146; XVSVault.sol (PR418-Base): 256~257, 910** | ● **Acknowledged** |

### 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.

#### PR 324

In the contract `JumpRateModelV2` , the role `DEFAULT_ADMIN_ROLE` of the `AccessControlManager` can grant addresses the privilege to call the following functions:

- `updateJumpRateModel()`

Any compromise to the `DEFAULT_ADMIN_ROLE` or accounts granted this privilege may allow a hacker to take advantage of this authority and do the following:

- Update the jump rate variables to increase or decrease the supply and borrow rates.

---

In the contract `RewardsDistributor` , the role `DEFAULT_ADMIN_ROLE` of the `AccessControlManager` can grant addresses the privilege to call the following functions:

- `setLastRewardingBlockTimestamp()`

Any compromise to the `DEFAULT_ADMIN_ROLE` or accounts granted this privilege may allow a hacker to take advantage of this authority and do the following:

- If the contract is time-based, set the last rewarding borrow and/or supply timestamps so that rewards are stopped earlier or later than expected.

---

#### PR 418

In the contract `XVSVault` , the role `DEFAULT_ADMIN_ROLE` of the `AccessControlManager` can grant addresses the privilege to call the following functions:

- `setRewardAmountPerBlockOrSecond()`

Any compromise to the `DEFAULT_ADMIN_ROLE` or accounts granted this privilege may allow a hacker to take advantage of this authority and do the following:

- Change the amount of reward tokens given our per second or block.

---

In the contract `XVSVault` , the role `admin` has authority over the following functions:

- `initializeTimeManager()`

Any compromise to the `admin` account may allow a hacker to take advantage of this authority and do the following:

- Initialize a contract to be time-based when it should be block-based, initialize a contract to be block-based when it should be time-based, or use the improper number of blocks in a year for the chain.

---

## PR 417

In the contract `RewardFacet` , the role `DEFAULT_ADMIN_ROLE` of the `AccessControlManager` can grant addresses the privilege to call the following functions:

- `seizeVenus()`

Any compromise to the `DEFAULT_ADMIN_ROLE` or accounts granted this privilege may allow a hacker to take advantage of this authority and do the following:

- Take any amount of `XVS` accrued by any account and send it to an address they control.

---

## PR 410

In the contract `SetterFacet` , the role `admin` has authority over the following functions:

- `_setXVSToken()`
- `_setXVSVToken()`

Any compromise to the `admin` account may allow a hacker to take advantage of this authority and do the following:

- Change the stored `XVS` token and vToken to addresses they control that contain malicious logic.

## 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, 01/12/2024]` :

We'll use the AccessControlManager (ACM) deployed at 0x4788629abc6cfca10f9f969efdeaa1cf70c23555.

In this ACM, only 0x939bd8d64c0a9583a7dcea9933f7b21697ab6396 (Normal Timelock) has the DEFAULT_ADMIN_ROLE. And this contract is a Timelock contract used during the Venus Improvement Proposals.

**PR 324**

The Normal timelock is already authorized to execute the function `updateJumpRateModel` , and it will be authorized to execute the function `setLastRewardingBlockTimestamp`

**PR 418**

We'll authorized Normal, [a] Fast-track and [b] Critical timelocks to execute `setRewardAmountPerBlockOrSecond`

The admin of the XVSVault is the Normal timelock (0x939bd8d64c0a9583a7dcea9933f7b21697ab6396), so the function `initializeTimeManager` is executable only via Governance

**PR 417**

We'll authorized Normal, [a] Fast-track and [b] Critical timelocks to execute seizeVenus

**PR 410**

The admin of the Unitroller contract (0xfD36E2c2a6789Db23113685031d7F16329158384) is the Normal timelock, so the mentioned functions will be executable only via Governance.

[a] 0x555ba73dB1b006F3f2C7dB7126d6e4343aDBce02

[b] 0x213c446ec11e45b15a6E29C1C1b402B8897f606d

The current config for the three Timelock contracts are:

normal: 24 hours voting + 48 hours delay
fast-track: 24 hours voting + 6 hours delay
critical: 6 hours voting + 1 hour delay

`[CertiK, 01/12/2024]` : The client has provided all steps towards mitigation on the BSC chain. In order to mitigate the finding completely, please provide the relevant information corresponding to the destination chains in which the contracts will initially be deployed.

## SSV-01 | BLOCK NUMBER-BASED VARIABLES SHOULD BE MADE IMMUTABLE

| Category | Severity | Location | Status |
|---|---|---|---|
| Design Issue | ● Minor | Shortfall/Shortfall.sol (PR324-Base): 73~74, 76~77, 99, 105 | ● Resolved |

### Description

Within the Shortfall contract, block-based defaults `DEFAULT_NEXT_BIDDER_BLOCK_LIMIT` and `DEFAULT_WAIT_FOR_FIRST_BIDDER` remain as 100 for the contract, even though a 100-block span may take drastically varying times depending on the chain. Since it is known that the time-based limits would be around 300 seconds, it may be beneficial to make these variables immutable so that a reasonable block limit value can be set based on the chain at the time of deployment.

### Recommendation

We recommend making the two variables cited above immutable in order to set them based on the chain in which the contract is deployed.

### Alleviation

`[CertiK, 01/16/2024]` : The client made changes resolving the finding in commits

- e1cdb75c1b3f9f9eaca720885704f75052eb69e3
- 1ab27b3fc7fc8d758197257dd76fcc709fa91c6a
- 408f0339207663b027c8af45690f9267a671af60

## VPB-03 | MAXIMUM SHOULD DEPEND ON CHAIN AND WHETHER THE CONTRACTS ARE TIME OR BLOCK-BASED

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | VToken.sol (PR324-Base): 759; VTokenInterfaces.sol (PR324-Base): 57~58 | ● Resolved |

### Description

The `MAX_BORROW_RATE_MANTISSA` is set to limit the borrow rate. However, the average block time of different chains will vary so that different maximums should be chosen. Additionally, if the contracts are time-based, then the maximum should be based on the maximum per second.

Considering if the contract is time-based with the current maximum of .0005% per second, this allows for an annual borrow rate of 15,768%.

### Recommendation

We recommend making the `MAX_BORROW_RATE_MANTISSA` an immutable variable that can be set to an appropriate value based on the chain it will be deployed on and whether it will be time or block-based.

### Alleviation

`[CertiK, 01/12/2024]` : The client made changes resolving the finding in the following commits

- ee4c083f24fe91bacbf4da10f2412dbeac476a53;
- 1ab27b3fc7fc8d758197257dd76fcc709fa91c6a.

## RFD-04 | MISSING OR INCOMPLETE NATSPEC

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency | ● Informational | RewardFacet.sol (PR417-Base): 140~144 | ● Resolved |

### Description

#### PR417 RewardFacet

- The comments above `seizeVenus()` do not include the return value.

### Recommendation

We recommend adding the missing or incomplete NatSpec comments mentioned above.

### Alleviation

`[CertiK, 01/12/2024]` : The client made the recommended changes in commit 7d2d183ab08543f4f06c33cb068232a58db35f02.

## VPB-01 | VALUE NAMING SHOULD BE UPDATED TO REFLECT POSSIBLE TIME-BASED CASE

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | ● Informational | Comptroller.sol (PR324-Base): 942; InterestRateModel.sol (PR324-Base): 10, 15, 25, 31; Lens/PoolLens.sol (PR324-Base): 507, 508, 511, 516, 536, 537, 539, 544; Rewards/RewardsDistributor.sol (PR324-Base): 22, 24, 69, 72, 78, 81, 146, 149, 448, 449, 450, 451; Shortfall/Shortfall.sol (PR324-Base): 353; VToken.sol (PR324-Base): 742~743, 806~807; VTokenInterfaces.sol (PR324-Base): 57, 131, 136, 285, 287, 287 | ● Resolved |

### Description

The locations cited contain local variables or comments which reference a value per block, and which are not modified to document the possibility that the averaged value may be *per block* or *per second*, according to choices made via the inherited `TimeManager` contract.

In the comments above `_setLastRewardingBlockTimestamp()` , the word "block" should be replaced with "block timestamp" since the function is for explicitly setting timestamp-related values.

### Recommendation

We recommend modifying the naming for the values above to accurately document that the time reference may be considered on a per block or per second basis.

### Alleviation

`[CertiK, 01/16/2024]` : The client made changes resolving the finding in commits 88265c396aaac27b24499a9c340dc2022aa762b5 and e16e4c7cf0283295602c38847a24f56a9d7900d1.

## VPH-01 | TYPOS AND INCONSISTENCIES

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency | ● Informational | Diamond/facets/FacetBase.sol (PR410-Base): 101; Diamond/facets/RewardFacet.sol (PR410-Base): 87, 104; Diamond/facets/SetterFacet.sol (PR410-Base): 584, 596; RewardFacet.sol (PR417-Base): 215~220 | ● Resolved |

## Description

### PR410 RewardFacet

- In the function `grantXVSInternal()` , the temporary variable `xvs` does not have a trailing underscore.
- In the function `grantXVSInternal()` the temporary variable `_xvsVToken` uses a leading underscore, however, according to the **Solidity Style Guide**, trailing underscores should be used to avoid naming collisions.

### PR417 RewardFacet

- The function `updateAndDistributeRewards()` is `internal` , however, it does not include Internal in the function name, which seems to be the convention throughout the codebase.

### PR410 SetterFacet

- The input of the function `_setXVSToken()` uses a leading underscore, when the convention followed throughout the codebase is to use a trailing underscore to avoid naming collisions for inputs.
- The input of the function `_setXVSVToken()` uses a leading underscore, when the convention followed throughout the codebase is to use a trailing underscore to avoid naming collisions for inputs.

### PR410 FacetBase

- In the function `releaseToVault()` the temporary variable `_xvs` uses a leading underscore, however, according to the **Solidity Style Guide**, trailing underscores should be used to avoid naming collisions.

## Recommendation

We recommend fixing the typos and inconsistencies mentioned above.

## Alleviation

`[CertiK, 01/12/2024]` : The client made changes resolving the finding in the following commits

- 78db5f88ba0e9a4c09286bbdea8687ac958e9813;
- 5032671039ea5ec95fc6dd63dbcec4d50b4212d3.

## VTT-01 | MISSING RETURN STATEMENT IN `setReduceReservesBlockDelta()` AND `setProtocolShareReserve()`

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Issue | ● Informational | VToken.sol (PR414-Base): 330~331, 341~342 | ● Resolved |

### Description

The function cited is intended to return uint type variables. However, neither function explicitly sets the return value within the function logic.

### Recommendation

We recommend including a return statement at the end or else removing the `returns (uint)` declaration for each function.

### Alleviation

`[CertiK, 01/12/2024]` : The client made changes resolving the finding in commit 9cfeba718e68aa7294c9895c51037f9e9b81e450.

## OPTIMIZATIONS | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| RFD-01 | Logic Can Be Skipped If Holder Has Zero Venus Accrued | Gas Optimization | Optimization | ● Resolved |
| RFD-02 | Unnecessary Variable Update | Gas Optimization | Optimization | ● Resolved |

## RFD-01 | LOGIC CAN BE SKIPPED IF HOLDER HAS ZERO VENUS ACCRUED

| Category | Severity | Location | Status |
|---|---|---|---|
| Gas Optimization | ● Optimization | RewardFacet.sol (PR417-Base): 154~157 | ● Resolved |

### Description

If `venusAccrued[holder] = 0` , then `totalHoldings` and `venusAccrued[holder]` will be updated unnecessarily.

### Recommendation

If this may be called on a holder such that `venusAccrued[holder]=0` , we recommend checking if `venusAccrued[holder] = 0` and skipping the unnecessary updates.

### Alleviation

`[CertiK, 01/12/2024]` : The client made the recommended changes in commit f0996f1e1f2c016e587ba14eb425f563543ebf00.

## RFD-02 | UNNECESSARY VARIABLE UPDATE

| Category | Severity | Location | Status |
|---|---|---|---|
| Gas Optimization | ● Optimization | RewardFacet.sol (PR417-Base): 185, 189 | ● Resolved |

### Description

In the function `claimVenus()` the temporary variable `j` is used as the index in a single for loop. However, it is unnecessarily set to zero as it will already be initialized to 0.

### Recommendation

We recommend removing the unnecessary setting of `j`.

### Alleviation

`[CertiK, 01/12/2024]` : The client made the recommended changes in commits

- 0c7e1f8ea0e3453c530dbcafa5be5849d62748ba;
- c86e591f7a037437b4dd6a024366b8d9535a6acf.

# APPENDIX | VENUS - TIME BASED CONTRACTS & SEIZEVENUS

## Finding Categories

| Categories | Description |
|---|---|
| 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. |
| 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. |
| Coding Issue | Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues. |
| 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.