



# Venus - E-mode and Liquidation Threshold on BNB Core Pool Security Assessment

CertiK Assessed on Sept 19th, 2025





Certik Assessed on Sept 19th, 2025

## Venus - E-mode and Liquidation Threshold on BNB Core Pool

The security assessment was prepared by Certik.

### Executive Summary

#### TYPES

Lending

#### ECOSYSTEM

Binance Smart Chain  
(BSC)

#### METHODS

Manual Review, Static Analysis

#### LANGUAGE

Solidity

#### TIMELINE

Preliminary comments published on 09/19/2025

Final report published on 09/19/2025

### Vulnerability Summary



12

Total Findings

8

Resolved

1

Timelock

0

Partially Resolved

3

Acknowledged

0

Declined



1 Centralization

1 Timelock



Centralization findings highlight privileged roles & functions and their capabilities, or instances where the project takes custody of users' assets.



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.



0 Major

Major risks may include logical errors that, under specific circumstances, could result in fund losses or loss of project control.



1 Medium

1 Resolved



Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.



2 Minor

1 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.



8 Informational

6 Resolved, 2 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 - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL

### **Summary**

[Executive Summary](#)

[Vulnerability Summary](#)

[Codebase](#)

[Audit Scope](#)

[Approach & Methods](#)

### **Overview**

[PR-614](#)

### **Dependencies**

[Third Party Dependencies](#)

[Assumptions](#)

[Recommendations](#)

### **Findings**

[VEC-01 : Centralization Related Risks](#)

[VEC-03 : Missing Input Validation](#)

[VEC-02 : Users May Switch Pools To Avoid Liquidation](#)

[VEC-12 : Missing Event Parameter](#)

[VEC-04 : Functions Removed From Interface](#)

[VEC-05 : Inconsistent Comments](#)

[VEC-06 : Inconsistent Naming Convention](#)

[VEC-07 : Inconsistent Use Of ``getCorePoolMarket\(\)``](#)

[VEC-08 : Discussion On Pool Handling](#)

[VEC-13 : Missing NatSpec Comments](#)

[VEM-02 : Discussion On ``getLiquidationParams\(\)``](#)

[VEM-03 : Implicit Change In ``getAccountLimits\(\)``](#)

### **Optimizations**

[VEC-09 : Redundant Checks](#)

[VEM-01 : Unnecessary Struct Element](#)

### **Appendix**

## **Disclaimer**

# CODEBASE | VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL

## Repository

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

## Commit

Base: [eda09f6b6a7bfecdcefae38a9c516a019379cfa4](#)

Update1: [87b206eb06f20b6ebbddb11fa53f928166aa69ae](#)

Update2: [23fae2f7a4bfde7ac1640bc3ae3588c8c9e2c8b7](#)

## Audit Scope

The file in scope is listed in the appendix.

## APPROACH & METHODS

# VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL

This audit was conducted for Venus to evaluate the security and correctness of the smart contracts associated with the Venus - E-mode and Liquidation Threshold on BNB Core Pool project. The assessment included a comprehensive review of the in-scope smart contracts. The audit was performed using a combination of Manual Review and Static Analysis.

The review process emphasized the following areas:

- Architecture review and threat modeling to understand systemic risks and identify design-level flaws.
- Identification of vulnerabilities through both common and edge-case attack vectors.
- Manual verification of contract logic to ensure alignment with intended design and business requirements.
- Dynamic testing to validate runtime behavior and assess execution risks.
- Assessment of code quality and maintainability, including adherence to current best practices and industry standards.

The audit resulted in findings categorized across multiple severity levels, from informational to critical. To enhance the project's security and long-term robustness, we recommend addressing the identified issues and considering the following general improvements:

- Improve code readability and maintainability by adopting a clean architectural pattern and modular design.
- Strengthen testing coverage, including unit and integration tests for key functionalities and edge cases.
- Maintain meaningful inline comments and documentations.
- Implement clear and transparent documentation for privileged roles and sensitive protocol operations.
- Regularly review and simulate contract behavior against newly emerging attack vectors.

# OVERVIEW

## VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL

This audit concerns the changes made in the in scope files outlined in the following PR:

- [PR-614](#)

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 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>.

### PR-614

PR-614 adds functionality to allow for e-mode (Efficiency Mode) pools. These enable pools with special collateral factors, liquidation thresholds, and liquidation incentives. For example, an e-mode pool could include highly correlated tokens and for that reason allow for higher collateral factors, higher liquidation thresholds, and lower liquidation incentives. A user may only belong to a single pool, however, a market may belong to multiple pools.

Part of these changes involve adding a liquidation threshold for each market and pool pair, which must be set higher than the collateral factor. The collateral factor gives the maximum value that a user can borrow, while the liquidation threshold gives the amount at which a borrow will become liquidateable. A user is able to supply assets in markets outside of the pool and if a pools `allowCorePoolFallback` is set to true it will count towards the users collateral value, in that case it will use the core pools collateral factor, liquidation threshold, and liquidation incentive for that market. In addition, a user is only able to borrow assets that are included in the pool and that have their borrows enabled. A user will not be able to enter a pool if they have borrows in a market that is not included in the pool or that does not have borrows enabled within the pool.

# DEPENDENCIES

## VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL

### 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:

- Oracles
- ERC20 Tokens

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.

### Assumptions

Within the scope of the audit, assumptions are made about the intended behavior of the protocol in order to inspect consequences based on those behaviors. Assumptions made within the scope of this audit include:

- The VIP that enables e-mode will set the `isBorrowAllowed` flag to true for all current markets in the core pool. Furthermore, when new markets are added to the core pool they will have this flag updated to be true if borrows are to be allowed.
- vBNB will neither be upgraded nor added to any e-mode groups.
- Venus will not support any tokens that charge a fee on transfer or utilize callbacks that may be used for reentrancy.

### 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. In addition, we recommend all assumptions about the behavior of the project are thoroughly reviewed and, if the assumptions do not match the intention of the protocol, documenting the intended behavior for review.



## FINDINGS

## VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL



12

Total Findings

0

Critical

1

Centralization

0

Major

1

Medium

2

Minor

8

Informational

This report has been prepared for Venus to identify potential vulnerabilities and security issues within the reviewed codebase. During the course of the audit, a total of 12 issues were identified. Leveraging a combination of Manual Review & Static Analysis the following findings were uncovered:

ID	Title	Category	Severity	Status
VEC-01	Centralization Related Risks	Centralization	Centralization	1h Timelock
VEC-03	Missing Input Validation	Logical Issue	Medium	Resolved
VEC-02	Users May Switch Pools To Avoid Liquidation	Logical Issue	Minor	Acknowledged
VEC-12	Missing Event Parameter	Logical Issue	Minor	Resolved
VEC-04	Functions Removed From Interface	Logical Issue	Informational	Acknowledged
VEC-05	Inconsistent Comments	Inconsistency	Informational	Resolved
VEC-06	Inconsistent Naming Convention	Coding Style	Informational	Acknowledged
VEC-07	Inconsistent Use Of <code>getCorePoolMarket()</code>	Inconsistency	Informational	Resolved
VEC-08	Discussion On Pool Handling	Logical Issue	Informational	Resolved
VEC-13	Missing NatSpec Comments	Inconsistency	Informational	Resolved

ID	Title		Category	Severity	Status
VEM-02	Discussion On	<code>getLiquidationParams()</code>	Design Issue	Informational	● Resolved
VEM-03	Implicit Change In	<code>getAccountLimits()</code>	Logical Issue	Informational	● Resolved

## VEC-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization	● Centralization	Comptroller/Diamond/facets/MarketFacet.sol (Base): 353, 396, 654; Comptroller/Diamond/facets/SetterFacet.sol (Base): 207, 221, 239, 254, 621	● 1h Timelock

### Description

Note that any centralization risks present in the existing codebase before the PR's in scope of this audit were not considered. Only those added to the in-scope PRs are addressed. We recommend all users 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>.

#### SetterFacet

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

- `setCollateralFactor(VToken, uint256, uint256)`
- `setLiquidationIncentive(address, uint256)`
- `setCollateralFactor(uint96,VToken, uint256, uint256)`
- `setLiquidationIncentive(uint96,address, uint256)`
- `setIsBorrowAllowed()`

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

- Set the collateral factor and liquidation threshold of any market in any pool to allow them to borrow more than should be allowed or cause users to become liquidateable when they should not be.
- Set the liquidation incentive for any market in any pool to receive more tokens than they should for liquidating or cause other liquidators to receive less tokens than they should for liquidating.
- Set the `isBorrowAllowed` bool in any market of any pool allowing them to disallow borrowing when it should be allowed or allow borrowing when it should be disallowed.

#### MarketFacet

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

- `createPool()` can create a new e-mode pool.
- `addPoolMarkets()` can add new markets to any pool except the core pool.
- `removePoolMarket()` can remove a market from any pool.

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

- Create a new e-mode pool.
- Add any new markets they want to an e-mode pool.
- Remove any market from any e-mode pool.

## 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 (2/3, 3/5) 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, 09/08/2025]: Only the Normal Timelock on BNB Chain has the DEFAULT\_ADMIN\_ROLE on the AccessControlManager contract.

Addresses:

- Normal Timelock: 0x939bD8d64c0A9583A7Dcea9933f7b21697ab6396
- Access Control Manager: 0x4788629abc6cfca10f9f969efdeaa1cf70c23555

So, only via a Normal VIP (involving the Venus Community) will be possible to grant permissions to execute the mentioned functions.

Normal, Fast-track and Critical Timelocks will have permissions to invoke the mentioned functions.

Addresses:

- FastTrack Timelock: 0x555ba73dB1b006F3f2C7dB7126d6e4343aDBce02
- Critical Timelock: 0x213c446ec11e45b15a6E29C1C1b402B8897f606d

## VEC-03 | Missing Input Validation

Category	Severity	Location	Status
Logical Issue	Medium	Comptroller/Diamond/facets/MarketFacet.sol (Base): 395, 490, 537, 685; Lens/VenusLens.sol (Base): 601	Resolved

### Description

#### MarketFacet

- The function `removePoolMarket()` does not check that the input `poolId` is not equal to the `corePoolId`. If the `corePoolId` is input, then it will cause modifications to the core pool, when the core pool should only be modified with the previous existing functionality.
- The function `getPoolVTokens()` references the `pools` mapping which is not updated for the core pool, however, it does not check that the input `poolId` is not equal to the `corePoolId`.
- The function `getLiquidationParams()` does not check that the input `vToken` is listed in the core pool.
- The function `poolMarkets()` does not check that the input `poolId` is a valid `poolId`.

#### VenusLens

- The function `getMarketsDataByPool()` does not check that the input `poolId` is not equal to the `corePoolId`, however, it references elements of the `pools` mapping which is not updated for the core pool. It also does not check that the input `poolId` is not greater than the `lastPoolId`.

### Recommendation

We recommend adding the checks mentioned above.

### Alleviation

[Venus, 09/08/2025]: We have added all the recommended checks except the `isListed` check in `getLiquidationParams`. This function is used in the account liquidity check, which previously did not validate `isListed`. To keep the behavior consistent, we will not add it here. Moreover, unlisted markets already have their risk parameters set to `0`, so they cannot impact.

[CertiK, 09/08/2025]: Considering the comments above and that the client made all other recommended changes in commit [e49c78380a7e03b17cb87e9bfb273e542e0f680b](#) we consider this finding resolved.

## VEC-02 | Users May Switch Pools To Avoid Liquidation

Category	Severity	Location	Status
Logical Issue	● Minor	Comptroller/Diamond/facets/MarketFacet.sol (Base): 334~338	● Acknowledged

### Description

The function `enterPool()` is called to enter a user into a new pool. The accounts liquidity is checked after changing the users poolId to ensure that the switch in pools would not cause the account to have insufficient liquidity. However, no liquidity check is performed prior to the switch in pools, which can allow a user who is liquidateable in one pool to switch to another pool that has higher collateral factors in order to cause their account to be non-liquidateable.

### Scenario

- Assume a user is entered in and borrowed in poolA, but is eligible for liquidation.
- Assume that all the user's borrows are allowed in `poolB` and `poolB` has higher collateral factors, so that if the user switches to `poolB` they would no longer be in shortfall.
- Prior to being liquidated the user switches to `poolB` in order to avoid liquidation.

This demonstrates how a user that is eligible for liquidation may be able to avoid it by switching pools.

### Recommendation

We recommend adding a check on the accounts liquidity in the current pool prior to switching pools.

### Alleviation

[Venus, 09/07/2025]: By design, we want to allow users the incentive to go to a pool saving them from liquidations. Its safe from a security perspective and only impact is the protocol loosing out on liquidation fees.

[CertiK, 09/08/2025]: Considering this is the intended design we have reduced the severity to minor and marking this finding as acknowledged.

## VEC-12 | Missing Event Parameter

Category	Severity	Location	Status
Logical Issue	● Minor	Comptroller/Diamond/facets/SetterFacet.sol (Base): 35~39	● Resolved

### Description

Liquidation incentives are now defined per market, so it should also emit the market whose liquidation incentive was updated.

### Recommendation

We recommend including an indexed `vToken` parameter in the `NewLiquidationIncentive` event.

### Alleviation

[CertiK, 09/08/2025]: The client made the recommended changes resolving the finding in commit [412322adff163d94b12a1f79a871faf62028e313](#).



## VEC-04 | Functions Removed From Interface

Category	Severity	Location	Status
Logical Issue	● Informational	Comptroller/Diamond/facets/SetterFacet.sol (Base): 196, 213	● Acknowledged

### Description

The functions `_setCollateralFactor()` and `_setLiquidationIncentive()` were removed from the `SetterFacet` and the `ISetterFacet` interface. While these are privileged functions, it is possible that some functionality depended on these functions as they were originally kept and had an alias added for them.

### Recommendation

We recommend ensuring that the removal of these functions does not cause any issues and that any functionality dependent on the functions was refactored. Furthermore, we recommend ensuring that their function selectors are removed from the facet mapping in the diamond proxy.

### Alleviation

[Venus, 09/08/2025]: Issue acknowledged. I won't make any changes for the current version.

Removing the functions `_setCollateralFactor()` and `_setLiquidationIncentive()` will not cause any issues, as they are only used through governance, and their selectors will be removed from the facet in the eMode configuration VIP. Since core pools now support LT, an additional argument is added, which changes the interface anyway. These functions are being removed to keep the interfaces consistent with the Isolated pools.

## VEC-05 | Inconsistent Comments

Category	Severity	Location	Status
Inconsistency	● Informational	Comptroller/ComptrollerStorage.sol (Base): 70, 304; Comptroller/Diamond/facets/FacetBase.sol (Base): 239, 256, 265; Comptroller/Diamond/facets/MarketFacet.sol (Base): 49, 151, 161, 186, 288, 440, 472; Comptroller/Diamond/facets/SetterFacet.sol (Base): 695; Lens/VenusLens.sol (Base): 112, 125, 575, 600	● Resolved

### Description

#### FacetBase

- The comments above the function `getPoolMarketIndex()` state "Returns the market index for a given vToken", which does not indicate that the index is dependent on the input `poolId`.
- The comments above the function `getCorePoolMarketIndex()` state "return The bytes32 key used to index into the `_poolMarkets` mapping for the Core Pool", which does not indicate that the key corresponds to the input `vToken`.
- The comments above the function `getCorePoolMarket()` state "return Market data corresponding to the given vToken", which does not indicate that the data is for the core pool.

#### SetterFacet

- The comments above the function `__setCollateralFactor()` state "Used by `_setCollateralFactor` and `setCollateralFactor`", however, `_setCollateralFactor` was removed. Furthermore, the comments do not include its ability to update the liquidation threshold.

#### MarketFacet

- The comments above the function `getEffectiveLiquidationIncentive()` state "This value should be used when calculating account liquidity and during liquidation checks." However, this does not correspond with how this value is used.
- The following functions comments do not clarify that they correspond to the core pool.
  - `getLiquidationIncentive()`
  - `_supportMarket()`
  - `unlistMarket()`
  - `isMarketListed()`
  - `checkMembership()`
  - `getAssetsIn()`

### ComptrollerStorage

- The comments above the mapping `accountMembership` do not state that it is only used in the core pool.
- The comments above the mapping `pools` do not state that it is only updated for e-mode pools. That is this mapping will always contain default values for the core pool.

### VenusLens

- The comments above the functions `getAllPoolsData()` and `getMarketsDataByPool()` do not clarify that they get e-mode pools data. That is they are not intended to fetch data for the core pool.
- The comments above the structs `PoolWithMarkets` and `MarketData` do not clarify they are for e-mode pools.

## Recommendation

We recommend fixing the typos and inconsistencies mentioned above.

## Alleviation

[CertiK, 09/08/2025]: The client made the recommended changes resolving this finding in commits

- [a03e12c8fab85db551d4b442b8169c57b8a0adba](#);
- [9881209b9caaea1f6e8952ba7f66041d8768766b](#);
- [2a01d66b2f05e5c7ce488f9027fc512cf94a5ce0](#).

## VEC-06 | Inconsistent Naming Convention

Category	Severity	Location	Status
Coding Style	● Informational	Comptroller/Diamond/facets/FacetBase.sol (Base): 258; Comptroller/Diamond/facets/SetterFacet.sol (Base): 202, 233, 701	● Acknowledged

### Description

The added code at times utilizes a leading underscore for private and internal functions/variables, while at some times it does not. We recommend determining the convention that should be followed for added internal/private functions/variables and ensuring the code conforms to it.

- `getCorePoolMarketIndex()` is `internal`, but does not have a leading underscore.
- `getCorePoolMarket()` is `internal`, but does not have a leading underscore.

Furthermore, the functions `setCollateralFactor()` and `__setCollateralFactor()` do not have a name that describes that they can also set the liquidation threshold.

### Recommendation

We recommend clarifying the intended convention and ensuring the code uniformly conforms to it.

### Alleviation

[Venus, 09/08/2025]: Issue acknowledged. I won't make any changes for the current version.

`getCorePoolMarketIndex()` and `getCorePoolMarket()` are internal functions without a leading underscore to stay consistent with other internal functions in the FacetBase. The leading underscore convention is only applied to internal functions that have an external wrapper with the same name.

`setCollateralFactor()` does not have a name that explicitly reflects its ability to also set the liquidation threshold, but this behavior is documented in the comments. This design choice was made to align with the Isolated Interface.

## VEC-07 | Inconsistent Use Of `getCorePoolMarket()`

Category	Severity	Location	Status
Inconsistency	● Informational	Comptroller/Diamond/facets/FacetBase.sol (Base): 177; Comptroller/Diamond/facets/MarketFacet.sol (Base): 157, 166, 194, 247, 640, 660, 661, 695, 701, 708; Comptroller/Diamond/facets/PolicyFacet.sol (Base): 327	● Resolved

### Description

The function `getCorePoolMarket()` returns the Market struct for a given `vToken` in the core pool and is used throughout the codebase.

However, often instead of using `getCorePoolMarket()`, the following code is used.

```
_poolMarkets[getCorePoolMarketIndex(address(vToken))]
```

### Recommendation

We recommend consistently using `getCorePoolMarket()` to improve the readability of the codebase.

### Alleviation

[CertiK, 09/08/2025]: The client made the recommended changes resolving this finding in commit [e9c96bcca3069630c488580a5f37bc2204cac1ea](#).

## VEC-08 | Discussion On Pool Handling

Category	Severity	Location	Status
Logical Issue	● Informational	Comptroller/Diamond/facets/SetterFacet.sol (Base): 620	● Resolved

### Description

Pools can have inconsistent states due to markets in the pool being removed or having their `isBorrowAllowed` bool set from `true` to `false`. This is because the user may have borrowed in the market, so that after the market is disallowed from borrowing they will have a borrow that should not be allowed in that pool. Can you please clarify in what scenarios it may be necessary to set a markets `isBorrowAllowed` bool from `true` to `false`. In particular, in such a case it may be desired to deprecate the pool and open a new e-mode pool with the new parameters to avoid users being in the pool with disallowed borrows.

In addition, if a market is removed from the pool, then any collateral supplied in that market will default back to the core pools collateral factor and liquidation threshold. As these will likely be lower than the e-mode pools it can cause a significant decrease in a users position potentially causing their position to become liquidateable. Users should be given sufficient notice in order to adjust their positions prior to such a change.

Furthermore, it may be desired to deprecate and prevent users from entering in an e-mode pool. In particular, the documentation provided stated that governance will be able to enable or disable entire e-mode groups. While all markets can be removed from the e-mode pool so that it reverts to the core pools values, it still may be desired to prevent users from entering the pool. In such a case checks can be added to ensure that users cannot call `enterPool()` with an input `poolId` that is no longer supported.

The provided documentation also stated that governance will be able to enable or disable markets as collateral/borrowable within e-mode groups. While borrowing can be disabled with the `isBorrowAllowed` flag, a user can supply and use any collateral within a pool. If that collateral is not included in the pool, it will use the core pools values, so it is not possible to disable the use of an asset as collateral without removing it from the core pool as well.

Lastly, originally the core pools never used the `isBorrowAllowed` bool and restricted borrowing in the markets via other methods. Furthermore, in the documentation provided it stated all core pools would have the bool set to true during the VIP. Considering this, all new core pool markets that are supported may want to be initialized with `isBorrowAllowed` set to true and having any borrows disallowed via the previous methods. This would allow for all core pools to remain consistent in the method that their borrows are disallowed.

### Recommendation

We recommend considering the comments above.

### Alleviation

[CertiK, 09/08/2025]: The client clarified the design intent and added an `isActive` bool which allows a single switch to

enable or disable e-mode pools via `setPoolAllowed()` in commits:

- [9eeef5a7ee6d515d15324c8cdedb6fc022bbe9df;](#)
- [ecf73ed8537d7c9f44a51af7d7c1faf989ea6085;](#)
- [fb1b0873e3c13999753822dbb51294dd4249ea83.](#)

## VEC-13 | Missing NatSpec Comments

Category	Severity	Location	Status
Inconsistency	● Informational	Comptroller/Diamond/facets/MarketFacet.sol (Base): 653; Comptroller/Diamond/facets/PolicyFacet.sol (Base): 520	● Resolved

### Description

#### PolicyFacet

- The function `poolBorrowAllowed()` does not have any NatSpec comments.

### Recommendation

We recommend adding the missing NatSpec comments mentioned above.

### Alleviation

[CertiK, 09/09/2025]: The client made the recommended changes in commit [5a9501fd1b5997337f86cbdae291240a348f4054](#).



## VEM-02 | Discussion On `getLiquidationParams()`

Category	Severity	Location	Status
Design Issue	● Informational	Comptroller/Diamond/facets/MarketFacet.sol (Update2): 728	● Resolved

### Description

The function `getLiquidationParams()` is used to get the `collateralFactorMantissa`, `liquidationThresholdMantissa`, and `liquidationIncentiveMantissa` for a market in a specific pool. It was adjusted so that in the case that a pool is active, the market is not listed in the pool, and the pool `allowCorePoolFallback` is set to false that it will not fallback to the core pool values and instead continue to use the returned `poolMarket` values.

In the current implementation, a market can only not be listed in a pool in two scenarios:

1. It was never listed in the pool.
2. It was listed in the pool, but was later unlisted via `removePoolMarket()`.

In both cases the `collateralFactorMantissa`, `liquidationThresholdMantissa`, `liquidationIncentiveMantissa` will be the default value of zero (because they were either never set or they were deleted to be reset to their default value).

In particular this updates the behavior when users who have collateral in markets that are not included in a pool whose `allowCorePoolFallback` is false. If they enter the pool their collateral in the markets not included in the pool will no longer contribute to their collateral value. It seems this is intended, because if users enter the pool, it will check if they are in shortfall accounting for this decrease in collateral value and revert if they would be in shortfall. So that this would effectively allow pools to disregard markets that it does not explicitly include from the liquidity calculations. However, it should be noted that typically if assets are not counted in collateral calculations they must not be entered in the market, that is their `accountMembership` in the core pool would be false. However, in this case users would still be entered in the market and their `accountMembership` in the core pool would still be true, which can cause issues as then the check in `seizeAllowed()`

```
if (!market.accountMembership[borrower]) {  
    return uint256(Error.MARKET_NOT_COLLATERAL);  
}
```

could be passed. If the account later becomes liquidateable then it is possible a liquidator would attempt to seize tokens from the market not included in the pool (whose liquidation incentive would be zero) resulting in them receiving nothing for the liquidation.

Furthermore, there may be issues with users who are entered in a market in a pool that is later unlisted. Of particular concern is the following scenario:

1. A market is listed in a pool whose `allowCorePoolFallback` is false.
2. A user enters the market (to allow it to count as collateral for borrows), enters in the pool, and then supplies collateral.
3. The user then borrows the maximum amount against it in another market in the pool that has its borrows allowed.

4. The market is unlisted from the pool but the pool is kept active and the `allowCorePoolFallback` is kept false.
5. Once it is unlisted the collateral factor and liquidation threshold become zero so that the user would be liquidatable, however, the liquidation incentive is zero so that users are not able to seize any tokens and the position will go unliquidated.

## Recommendation

We recommend considering the items above and clarifying the intended behavior.

## Alleviation

[Venus, 09/19/2025]: This is an intended behavior. If a user supplies an asset that is not being used as collateral in eMode and does not exit those markets while remaining in an eMode, those assets stay at risk of liquidation. This is consistent with the core pool, where entering a market with a collateral factor set to 0 still leaves the asset subject to liquidation risk.

The UI could display a warning when users enter eMode, and the documentation will recommend that users exit unused markets to avoid unnecessary exposure.

On the liquidator side, we will also recommend checking incentives before initiating liquidation.

Keeping the liquidation incentive at 0 for these unused markets is reasonable-it discourages liquidations, reinforces the intended behavior, and ensures liquidations only occur when a user's account is actually liquidatable. If liquidators still choose to act, it ultimately helps restore the user's account to a healthier state, so users remain responsible for managing their own risk.

Markets will only be unlisted from a pool after a full user risk analysis and with Chaos Labs' recommendations, executed through governance. If the goal is to control liquidations rather than immediately delist, we can first set the collateral factor (CF) and liquidation threshold (LT) to 0 (while maintaining any required LT). The market will then be delisted only once the risk analysis confirms that it is safe to proceed.

## VEM-03 | Implicit Change In `getAccountLimits()`

Category	Severity	Location	Status
Logical Issue	● Informational	Comptroller/Diamond/facets/PolicyFacet.sol (Update2): 432~441	● Resolved

### Description

The function `getAccountLiquidity()` was adjusted to use the liquidation threshold instead of the collateral factor. As such any functionality that depends on this will have their behavior updated as well. In particular, the function

`getAccountLimits()` will now return limits according to the liquidation threshold as opposed to the collateral factor.

### Recommendation

We recommend ensuring that `getAccountLimits()` is consistent with the intended behavior. If the intended behavior is to base the limits on the collateral factor it can instead call `getBorrowingPower()`. Note that the introduction of the liquidation threshold may break users interactions with functions depending on how they interpreted the original stand alone collateral factor (either as the limit for liquidation or the limit for borrowing). In general, we recommend ensuring users are aware of the potential breaking changes and have time to conform to them.

### Alleviation

[Venus, 09/19/2025]: Originally, the core pool didn't have a liquidation threshold (LT); only the collateral factor (CF) was used. In `getAccountLiquidity`, and by extension in `getAccountLimits`, CF was previously used to determine the liquidation limits, but now LT serves that role. We've already updated `getAccountLiquidity` to use LT instead of CF, so it makes sense to keep relying on `getAccountLiquidity` for `getAccountLimits` to ensure the returned values remain consistent with their original purpose - i.e., reflecting liquidation limits rather than borrowing limits. Moreover, this function isn't currently used in the protocol's UI, and we'll make sure to highlight this change in the documentation so that users are aware.

## OPTIMIZATIONS

VENUS - E-MODE AND LIQUIDATION THRESHOLD  
ON BNB CORE POOL

ID	Title	Category	Severity	Status
VEC-09	Redundant Checks	Code Optimization	Optimization	● Resolved
VEM-01	Unnecessary Struct Element	Code Optimization	Optimization	● Acknowledged

## VEC-09 | Redundant Checks

Category	Severity	Location	Status
Code Optimization	● Optimization	Comptroller/Diamond/facets/MarketFacet.sol (Base): 654	● Resolved

### Description

`_addPoolMarket()` is internal and only called by `addPoolMarkets()`, which will repeatedly call `ensureAllowed("addPoolMarket(uint96,address)")`. If the check passes for the first call, it will pass for all subsequent calls and is unnecessary.

### Recommendation

We recommend restricting access to call the external function `addPoolMarkets()` and removing the access restriction on the internal function `_addPoolMarket()`.

### Alleviation

[Certik, 09/08/2025]: The client made the recommended changes resolving this finding in commit [5d4b37745f83a05f8369617dfe00a2f453f0a32d](#).

## VEM-01 | Unnecessary Struct Element

Category	Severity	Location	Status
Code Optimization	● Optimization	Lens/VenusLens.sol (Update2): 167	● Acknowledged

### Description

The element `isVenus` in the `MarketsInfo` is unused and can be removed. The `MarketsInfo` struct is only used in the function `vTokenMetadata()` for the return values of the `markets()` call. However, the return value `isVenus` is unused so that it can simply be skipped and the corresponding element removed from the struct.

### Recommendation

We recommend removing the `isVenus` of the `MarketsInfo` struct.

### Alleviation

[Venus, 09/19/2025]: The new `MarketsInfo` struct was introduced to hold data from the comptroller's `_poolMarkets` mapping, helping reduce the number of local variables. While the `isVenus` flag is not being actively used at the moment, it remains part of the mapping and could be useful for future requirements. For that reason, we prefer to keep the `isVenus` element in the struct rather than removing it now.

## APPENDIX

# VENUS - E-MODE AND LIQUIDATION THRESHOLD ON BNB CORE POOL


### Audit Scope

VenusProtocol/venus-protocol

- Comptroller/Diamond/facets/FacetBase.sol
- Comptroller/Diamond/facets/MarketFacet.sol
- Comptroller/Diamond/facets/SetterFacet.sol
- Comptroller/Diamond/facets/PolicyFacet.sol
- Comptroller/ComptrollerStorage.sol
- Lens/VenusLens.sol
- Comptroller/Diamond/facets/RewardFacet.sol
- Comptroller/Diamond/interfaces/IFacetBase.sol
- Comptroller/Diamond/interfaces/IMarketFacet.sol
- Comptroller/Diamond/interfaces/ISetterFacet.sol
- Comptroller/Diamond/Diamond.sol
- Comptroller/ComptrollerInterface.sol
- Comptroller/ComptrollerLensInterface.sol
- Comptroller/Types/PoolMarketId.sol
- InterfacesV8.sol
- Lens/ComptrollerLens.sol
- Liquidator/Liquidator.sol
- Tokens/VAI/VAIController.sol

VenusProtocol/venus-protocol

 Tokens/VTokens/VToken.sol

 Utils/ErrorReporter.sol

## 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.



## DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR

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.

# Elevating Your **Web3** Journey

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is the largest 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.



Venus - E-mode and Liquidation Threshold on BNB Core Pool  
Security Assessment

CertiK Assessed on Sept  
19th, 2025

Copyright ©  
CertiK