



Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Auxiliary Recovery Contract	Documentation quality	High	<div><div></div></div>
Timeline	2025-05-08 through 2025-05-08	Test quality	Medium	<div><div></div></div>
Language	Solidity	Total Findings	4	<div><div></div><div>Fixed: 1</div><div>Acknowledged: 3</div></div>
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review	High severity findings ⓘ	0	
Specification	Internal Documentation	Medium severity findings ⓘ	1	<div><div></div><div>Acknowledged: 1</div></div>
Source Code	<ul style="list-style-type: none">VenusProtocol/isolated-pools #74ed2f2 	Low severity findings ⓘ	2	<div><div></div><div>Fixed: 1</div><div>Acknowledged: 1</div></div>
Auditors	<ul style="list-style-type: none">Nikita Belenkov Senior Auditing EngineerHytham Farah Auditing EngineerCameron Biniamow Auditing Engineer	Undetermined severity findings ⓘ	0	
		Informational findings ⓘ	1	<div><div></div><div>Acknowledged: 1</div></div>

Summary of Findings

Venus Protocol suffered a wUSDM donation attack on their ZkSync deployment in April 2025. The attacker (Account 1) and several other accounts (Accounts 2-5) exploited a price manipulation vulnerability, resulting in approximately \$915K of bad debt across multiple accounts:

1. Account 1 (attacker): \$861K in wUSDM debt without collateral.

2. Accounts 2-5: Various positions with approximately \$54K in net bad debt.

The attack drained liquidity from the wUSDM market, creating significant imbalances between collateral and borrowing positions. The Venus team developed a specialized recovery contract called `WUSDMLiquidator` to partially restore market stability through a carefully sequenced transaction that:

1. Injects \$400K in wUSDM liquidity from the Venus Treasury.

2. Temporarily modifies protocol parameters to:

◦ Increase the Close Factor to 100%.

◦ Set the minimum liquidation collateral amount to 0.

◦ Adjust the collateral factor for vwUSDM.

3. Executes a precise liquidation sequence to:

◦ Borrow the debt assets (USDT, USDC.e, and WETH) using the \$400K wUSDM collateral.

◦ Liquidate all debts across Accounts 2-5.

◦ Capture seized wUSDM collateral from Accounts 2-5.

4. Restores original protocol parameters once recovery is complete.

The solution specifically prevents third-party liquidators from seizing the newly injected liquidity by performing all actions atomically in a single transaction. This recovery effort deliberately leaves Account 1's \$861K bad debt unaddressed, as addressing it would require additional funds beyond the \$400K allocated for this operation. After the recovery, the `WUSDMLiquidator` contract will maintain a controlled debt position with a healthy collateralization ratio.

The audit has revealed only one medium severity issue where the Oracle price is not verified. This should be addressed before deployment.

Fix Review Update

All issues have been acknowledged with a reasonable explanation. During the fix review, the Venus team identified `VEN-3`, which has been added to the report and fixed.

ID	DESCRIPTION	SEVERITY	STATUS
VEN-1	Missing Validation on Oracle Price Retrieval	• Medium ⓘ	Acknowledged
VEN-2	Unchecked Return Values From External Calls	• Low ⓘ	Acknowledged
VEN-3	Collateral Insufficiency in WUSDMLiquidator Prevents Full Liquidation of A2	• Low ⓘ	Fixed
VEN-4	Hardcoded Liquidation Incentive	• Informational ⓘ	Acknowledged

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

i **Disclaimer**

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
 1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
 2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
 1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
 2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

Files Included

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

Included Paths: `contracts/WUSDMLiquidator.sol` , `contracts/ComptrollerStorage.sol`

Operational Considerations

1. The Venus `Comptroller` contract must temporarily authorize the `WUSDMLiquidator` contract to:
- Update the minimum liquidation collateral amount.
 - Set the close factor.
 - Set the collateral factor for `vwUSDM`.
 - Pause and unpause the `MINT` and `ENTER_MARKET` actions for the `vwUSDM` market.
 - Set the protocol seize share amount for the `vwUSDM`, `vWETH`, `vUSDCe`, and `vUSDT` markets.
- Following the execution of the liquidations, the Venus team should revoke all authorizations granted to the `WUSDMLiquidator` contract.
2. While the `WUSDMLiquidator` contract may have a sufficient collateralization ratio immediately following the liquidations, price fluctuations of the debt assets may reduce the collateralization ratio to a value below the liquidation threshold. Thus, the `WUSDLiquidator` contract could be liquidated, further losing funds. The Venus team must monitor the collateralization ratio of the `WUSDLiquidator` contract and act accordingly to prevent unnecessary liquidations.
3. The Comptroller upgrade and restoration process is managed separately from the `WUSDMLiquidator`'s execution flow. While the liquidator contract handles the actual liquidation of underwater positions, the prerequisite system modifications (upgrading the `Comptroller` and later restoring it) will be executed manually by the Venus Protocol team through separate governance transactions via the `NormalTimelock`. These actions require their own governance approval and are not directly part of the liquidator's automated functionality.
4. The `WUSDMLiquidator` contract operates under the assumption that the required 400,000 `wUSDM` tokens have already been transferred to it before execution. The contract does not explicitly verify this specific amount, but instead simply uses whatever `wUSDM` balance is available in the contract at runtime through the `WUSDM.balanceOf(address(this))` call. The transfer of these tokens to the liquidator contract is a prerequisite step that will be handled separately by the Venus Protocol team prior to triggering the liquidation process.
5. The `NormalTimelock` must call `acceptOwnership()` before it can execute the `run()` function, as the deployment only initiates the first step of the ownership transfer process. Without this step, the recovery operation cannot proceed.

Key Actors And Their Capabilities

The `WUSDMLiquidator` contract will be transferred to the control of the Venus Protocol governance through the `NormalTimelock` contract, placing the recovery process entirely in the hands of the community. Upon deployment, ownership is immediately transferred to the timelock, according to the deployment script in the audited PR. This ensures that the execution of the recovery logic can only be triggered through a formal governance proposal and vote. This governance-controlled approach guarantees that the complex liquidation and debt repayment operations are executed with community oversight and approval, rather than at the discretion of any individual actor.

Deployer

Responsibility

- Deploy the `WUSDMLiquidator` contract and ensure the transition of control to protocol governance.

Trust Assumption

- Will deploy according to the script in the PR which transfers ownership to community governance.

Post-Recovery Debt Manager

Responsibility

- Manage the debt position of `WUSDMLiquidator` after the recovery process is complete.

Trust Assumption

- Holds delegated authority solely for managing debt, not ownership or governance.
- Will reduce debt as soon as more `wUSDM` becomes available.

Findings

VEN-1 Missing Validation on Oracle Price Retrieval

● Medium ⓘ

Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

- The `wUSDM` price oracle we used is based on two components:
1. Exchange rate `wUSDM/USDM`, read from the `wUSDM` contract each time. This exchange rate can be "artificially" increased, with a donation, for example. But only privileged accounts can "artificially" decrease it.
 2. `USDM/USD` ratio, provided by Chainlink. This feed shouldn't be manipulable in one transaction

File(s) affected: `WUSDMLiquidator.sol`

Description: In `WUSDMLiquidator.sol`, the `run()` function retrieves the `wUSDM` price from the oracle:

```
uint256 wusdmPrice = ORACLE.getPrice(address(WUSDM));
```

There is no validation on this price. If the oracle returns `0` or a manipulated value, liquidation logic may act on faulty data. The risk is heightened due to the `OmnichainGovernanceExecutor`, which allows **anyone** to execute approved proposals after a delay.

In the similar manner in `_getBorrowedTokensToCollateralVTokensRatio()` makes a call to `ORACLE.getUnderlyingPrice()`, which should also be validated to avoid a manipulated value.

Exploit Scenario:

An example exploit scenario with wUSDM where an attacker could:

- Use a flash loan to manipulate liquidity pools feeding the oracle
- Force the wUSDM price to near zero
- Call `execute()` on `OmnichainGovernanceExecutor` to trigger `run()`
- Disrupt the liquidation logic with a skewed value of wUSDM or one of the vTokens being liquidated.

This can be done within a single atomic transaction once the timelock expires. Note that such an attack would not personally benefit the attacker, rather, it would only harm the Venus protocol.

In a similar manner, the other tokens where price is verified can be used as an attack vector.

Recommendation: Validate the oracle price before using it.

VEN-2 Unchecked Return Values From External Calls

• Low ⓘ Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

We have simulated the execution of the VIP, and the Venus contracts will revert the TX if the status is not the expected one

File(s) affected: `WUSDMLiquidator.sol`

Description: The contract calls the `mint()`, `borrow()`, `liquidateBorrow()`, and `repayBorrowBehalf()` Venus protocol functions that return error codes instead of reverting on failure.

These return a `uint256` error code (0 = `NO_ERROR` on success), but the current implementation ignores the return values. If a call fails silently (non-zero return), subsequent logic may proceed under false assumptions, leading to incorrect state or undercollateralization.

Recommendation: Check and validate return values from all Venus protocol calls.

VEN-3 Collateral Insufficiency in `WUSDMLiquidator` Prevents Full Liquidation of A2

• Low ⓘ Fixed

i Update

Marked as "Fixed" by the client.
Fixed in commit `282d2088fb04c014460bcc07fefbcc3b24481218`.

File(s) affected: `contracts/WUSDMLiquidator.sol`

Description: The `WUSDMLiquidator` contract is initially supplied with approximately 370k wUSDM, which is used as collateral to borrow vWETH and liquidate the full debt of A2 (approximately 160 vWETH). However, there is no consideration of whether the value of the supplied wUSDM collateral is sufficient to borrow the full vWETH debt amount for A2. Due to recent increases in the price of ETH and the fixed amount of wUSDM collateral, the contract is unable to borrow the entire vWETH debt amount for A2.

Recommendation: Update the liquidation logic in `WUSDMLiquidator` so that it calculates the exact amount required to seize 100% of A2's wUSDM collateral, based on the market prices. Note that this will result in A2 maintaining some amount of bad debt (the difference between the value of A2's vWETH debt and wUSDM collateral).

VEN-4 Hardcoded Liquidation Incentive

• Informational ⓘ Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

The value returned by `Comptroller.liquidationIncentiveMantissa()` can be updated only with a VIP. We'll monitor the proposed VIP before the proposal of the current one, to avoid any discrepancy

File(s) affected: `WUSDMLiquidator.sol`

Description: `WUSDMLiquidator` hardcodes `LIQUIDATION_INCENTIVE = 1.1e18` instead of fetching it from `Comptroller.liquidationIncentiveMantissa()`, risking inconsistencies if the protocol updates this value.

Currently, the two values match.

Recommendation: Replace the constant with a dynamic fetch from the Comptroller.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Files

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

- `cc0...7dc ./contracts/ComptrollerStorage.sol`
- `0d2...0ed ./contracts/WUSDMLiquidator.sol`

Test Suite Results

Tests based on the forked state of zkSync are present and all pass.

```
Forking test on block of 59859399
Ran 2 tests for test/Liquidator.t.sol:LiquidatorTest
[PASS] testGetBeaconAddress() (gas: 9179)
Logs:
Beacon address: 0x0221415aF47FD261dD39B72018423dADe5d937c5
[PASS] testRun() (gas: 16257247)
Logs:
=== check state before ===
liquidity of A2 is: 0
shortfall of A2 is: 60813704974064221501532
vwUSDM balance of A2 is: 27600971020155
Market vweth debt of A2 is:159344103812547681617
liquidity of A3 is: 0
```



```
shortfall of A3 is: 33045687714775514046380
vwUSDM balance of A3 is: 4929412019494
Market vusdt debt of A3 is:55253303744
Market vusdce debt of A3 is:19388753561
liquidity of A4 is: 0
shortfall of A4 is: 26936230662235911869497
vwUSDM balance of A4 is: 3528279214003
Market vusdt debt of A4 is:21631131316
Market vusdce debt of A4 is:35072750180
liquidity of A5 is: 0
shortfall of A5 is: 10888333336513344512552
vwUSDM balance of A5 is: 15966710
Market vusdt debt of A5 is:8366180527
Market vusdce debt of A5 is:2521082990
WETH debt of a2 before repay: 12081574383012029517
USDC debt of a3 before repay: 1
USDC debt of a4 before repay: 374953358
USDC debt of a5 before repay: 2520925971
USDT debt of a3 before repay: 26167701080
USDT debt of a4 before repay: 21631131316
USDT debt of a5 before repay: 8366180527
=== check state after ===
liquidity of A2 is: 16876253176
shortfall of A2 is: 0
vwUSDM balance of A2 is: 2
Market vweth debt of A2 is:1
liquidity of A3 is: 0
shortfall of A3 is: 1266058806663
vwUSDM balance of A3 is: 87
Market vusdt debt of A3 is:1
Market vusdce debt of A3 is:1
liquidity of A4 is: 0
shortfall of A4 is: 1671088927125
vwUSDM balance of A4 is: 39
Market vusdt debt of A4 is:1
Market vusdce debt of A4 is:1
liquidity of A5 is: 0
shortfall of A5 is: 1316687571721
vwUSDM balance of A5 is: 81
Market vusdt debt of A5 is:1
Market vusdce debt of A5 is:1
Liquidity of liquidator is: 179871760446603873565890
Shortfall of liquidator is: 0
vwUSDM balance of liquidator is: 72982327972910
WETH debt of liquidator: 159344103812547681617
USDC.e debt of liquidator: 56982586731
USDT debt of liquidator: 85250615587
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 309.44s (286.59s CPU time)
Ran 1 test suite in 309.99s (309.44s CPU time): 2 tests passed, 0 failed, 0 skipped (2
total tests)
```

Changelog

- 2025-05-08 - Initial report
- 2025-05-22 - Final report

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp’s mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp’s team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over \$200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked

with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

