Quantstamp

Venus - Correlated Oracles

Executive Summary

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

Type	Oracle System
Timeline	2024-04-01 through 2024-04-08
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Internal Documentation
Source Code	• VenusProtocol/oracle #3e86e32
Auditors	• Julio Aguilar Auditing Engineer • Mustafa Hasan Senior Auditing Engineer • Nikita Belenkov Auditing Engineer

Documentation quality	Medium
Test quality	High
Total Findings	4 Fixed: 2 Acknowledged: 2
High severity findings ⓘ	0
Medium severity findings ⓘ	0
Low severity findings ⓘ	2 Fixed: 1 Acknowledged: 1
Undetermined severity findings ⓘ	0
Informational findings ⓘ	2 Fixed: 1 Acknowledged: 1

Summary of Findings

This audit covered an oracle system used in the wider Venus ecosystem. This oracle system is designed to work with correlated tokens and return the USD price of the underlying token amount for one correlated token via a mix of native token contracts and staking managers.

The audit has not revealed any significant issues, only a few Low- and informational-severity Issues that should still be addressed before contracts are deployed. The audit report also contains the VER-COR-2 issue that the Venus team reported to us.

Overall, the code is well-written and documented. The test suite consists of 128 tests with 92.16% coverage.

Fix Review

All issues have either been fixed or acknowledged.

ID	DESCRIPTION	SEVERITY	STATUS
VEN-COR-1	**Inconsistency in the Function Call Sequence Could Lead to an Outdated Price**	● Low ⓘ	Acknowledged
VEN-COR-2	`OneJumpOracle` **Must Return a Value Scaled to the Underlying Asset Decimals**	● Low ⓘ	Fixed
VEN-COR-3	**Upgradable Tokens Can Have Their Functionality Altered Which May Result in Inaccurate Rates**	● Informational ⓘ	Acknowledged
VEN-COR-4	**Outdated Solidity Version**	● Informational ⓘ	Fixed

Assessment Breakdown

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

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

Audit Scope

- `contracts/oracles/AnkrBNBOracle.sol`
- `contracts/oracles/BNBxOracle.sol`
- `contracts/oracles/OneJumpOracle.sol`
- `contracts/oracles/PendleOracle.sol`
- `contracts/oracles/SFraxOracle.sol`
- `contracts/oracles/SFrxETHOracle.sol`
- `contracts/oracles/SlisBNBOracle.sol`
- `contracts/oracles/StkBNBOracle.sol`
- `contracts/oracles/WBETHOracle.sol`
- `contracts/oracles/WeETHOracle.sol`
- `contracts/oracles/WstETHOracle.sol`
- `contracts/oracles/common/CorrelatedTokenOracle.sol`

Findings

VEN-COR-1

Inconsistency in the Function Call Sequence Could Lead to an Outdated Price

● Low ⓘ Acknowledged

Update

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

> We don’t have TWAPOracle anymore therefore right now there is no use of the updateAssetPrice function. We’ll upgrade the updateAssetPrice in the future to update the underlying associated with a correlated asset in a transparent way.

File(s) affected: `oracles/common/CorrelatedTokenOracle.sol`

Description: According to the NatSpec comments of the Resilient oracle, the function `updateAssetPrice()` should be called before calling `getPrice()`. However, the `CorrelatedTokenOracle` calls the latter without calling the former which could result in an outdated price from the pivot oracle inside the Resilient oracle.

Recommendation: Make sure to call `updateAssetPrice()` before calling `getPrice()`.

VEN-COR-2

`OneJumpOracle` Must Return a Value Scaled to the Underlying Asset Decimals

● Low ⓘ Fixed

> **Update**
>
> The return value is now correctly scaled. Fixed in commits `9efcae374301d2218e1f53e3f3df638f35817241` and `eeab989a9c6e2f44555cba629b944fb99835f193`.

File(s) affected: `contracts/oracles/OneJumpOracle.sol`

Description: `OneJumpOracle` uses `INTERMEDIATE_ORACLE` oracles that follow the Compound oracle convention, which means that they return values that are scaled to `36 - correlatedDecimals`. This is incompatible with the requirement from the `CorrelatedTokenOracle`, where it expects the `OneJumpOracle` to return a value scaled to the underlying asset decimals.

Recommendation: Scale the `OneJumpOracle` return value to the underlying asset decimals.

VEN-COR-3

Upgradable Tokens Can Have Their Functionality Altered Which May Result in Inaccurate Rates

● Informational ⓘ Acknowledged

> **Update**
>
> Marked as "Acknowledged" by the client. The client provided the following explanation:
>
> > We'll update the documentation site with this info

File(s) affected: `oracles/BNBxOracle.sol`, `oracles/SlisBNBOracle.sol`, `oracles/StkBNBOracle.sol`, `oracles/WBETHOracle.sol`, `oracles/WeETHOracle.sol`, `oracles/AnkrBNBOracle.sol`

Description: The above mentioned custom oracles rely on a price feed that has an upgradable component behind it, meaning at some point, the implementation of the price feed used can change, and hence, such an oracle can start producing incorrect or inaccurate prices. These upgrade actions should be carefully monitored, and adjustments should be made to the custom oracle if needed.

Recommendation: This risk should be documented to the users and upgrades carefully monitored.

VEN-COR-4 Outdated Solidity Version

● Informational ⓘ Fixed

> **Update**
>
> The Solidity version was upgraded to `0.8.25`. Fixed in: `752da30d752f0df45f666778a2ff65b4040bed67`.

File(s) affected: `All in-scope contracts`

Description: The project uses Solidity version `0.8.13`, which is not recommended for deployment.

Recommendation: We recommend using the latest version `0.8.25`.

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).

Adherence to Best Practices

1. Acknowledged `SFrxETHOracle` can call `pricePerShare()` instead of `convertToAssets(amount)`, which is a function made to get the amount of the frxETH for 1 sfrxETH.
2. Acknowledged Similarly, WstETHOracle can call stEthPerToken() instead of `getPooledEthByShares(amount)`, which is a function made to get the amount of the stETH for 1 wstETH.
3. Fixed `OracleInterface` is imported in `StkBNBOracle.sol`, however it is not used and should be removed.
4. Fixed `CorrelatedTokenOracle.getUnderlyingAmount()` should have an underscore before its name as it is an internal function.

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- Slither v0.10.0

Steps taken to run the tools:

1. Install the Slither tool: `pip3 install slither-analyzer`
2. Run Slither from the project directory: `slither .`

Automated Analysis

Slither

401 results were found across the codebase. Non-false positive findings have been included in the report.

Test Suite Results

The test suite has a total of 128 tests of which all pass.

```
AnkrBNBOracle unit tests
  deployment
    ✔ revert if ankrBNB address is 0
    ✔ revert if ResilientOracle address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if ankrBNB address is wrong
    ✔ should get correct price

BNBxOracle unit tests
  deployment
    ✔ revert if stakeManager address is 0
    ✔ revert if BNBx address is 0
    ✔ revert if resilientOracle address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if BNBx address is wrong
    ✔ should get correct price

Binance Oracle unit tests
  ✔ set price
  ✔ set BNB price
  ✔ fetch price (46ms)
  ✔ fetch BNB price
  ✔ price expired (56ms)
  ✔ set WBETH price
  ✔ fetch WBETH price
  ✔ revert when setting feed registry address and sid already available
  ✔ revert when feed registry address is zero (65ms)
  ✔ fetch price from direct feed registry  (51ms)

bound validator
  add validation config
    ✔ length check
    ✔ validation config check
    ✔ config added successfully & event check
  validate price
    ✔ validate price (99ms)

Oracle unit tests
  set token config
    ✔ cannot set feed to zero address
    ✔ sets a token config
  batch set token configs
    ✔ cannot set feed or vtoken to zero address
    ✔ parameter length check
    ✔ set multiple feeds
  getPrice
    ✔ gets the price from Chainlink for vBNB
    ✔ gets the price from Chainlink for USDC
    ✔ gets the price from Chainlink for USDT
    ✔ gets the price from Chainlink for DAI
    ✔ gets the direct price of a set asset
    ✔ reverts if no price or feed has been set
  setDirectPrice
    ✔ sets the direct price
  stale price validation
    ✔ stale price period cannot be 0
    ✔ modify stale price period will emit an event
    ✔ revert when price stale
    ✔ if updatedAt is some time in the future, revert it
    ✔ the chainlink anwser is 0, revert it

OneJumpOracle unit tests
  deployment
    ✔ revert if correlated token address is 0
    ✔ revert if underlying token address is 0
    ✔ revert if resilient oracle address is 0
    ✔ revert if intermediate oracle address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if address is not valid LDO address
    ✔ should get correct price of LDO

WstETHOracle unit tests
  deployment
    ✔ revert if market address is 0
    ✔ revert if ptOracle address is 0
    ✔ revert if ptWeETH address is 0
    ✔ revert if eETH address is 0
    ✔ revert if ResilientOracle address is 0
    ✔ revert if TWAP duration is 0
    ✔ revert if invalid TWAP duration
    ✔ should deploy contract
  getPrice
    ✔ revert if wstETH address is wrong
    ✔ should get correct price

Oracle plugin frame unit tests
```

```
  admin check
    ✔ transfer owner
  token config
    add single token config
      ✔ token can"t be zero & maxStalePeriod can't be zero
      ✔ token config added successfully & events check
    batch add token configs
      ✔ length check
      ✔ token config added successfully & data check
  get underlying price
    ✔ revert when asset not exist
    ✔ revert when price is expired
    ✔ revert when price is not positive (just in case Pyth return insane data) (45ms)
    ✔ price should be 18 decimals (75ms)
  validation
    ✔ validate price (118ms)
    ✔ validate BNB price (90ms)

Oracle plugin frame unit tests
  token config
    add single token config
      ✔ vToken can"t be zero & main oracle can't be zero
      ✔ reset token config (72ms)
      ✔ token config added successfully & events check (54ms)
    batch add token configs
      ✔ length check
      ✔ token config added successfully & data check (159ms)
  change oracle
    set oracle
      ✔ null check (75ms)
      ✔ existance check
      ✔ oracle set successfully & data check (66ms)
  get underlying price
    ✔ revert when protocol paused (38ms)
    ✔ revert price when main oracle is disabled and there is no fallback oracle
    ✔ revert price main oracle returns 0 and there is no fallback oracle
    ✔ revert if price fails checking
    ✔ check price with/without pivot oracle (44ms)
    ✔ disable pivot oracle
    ✔ enable fallback oracle (88ms)
    ✔ Return fallback price when fallback price is validated successfully with pivot oracle
    ✔ Return main price when fallback price validation failed with pivot oracle

SFraxOracle unit tests
  deployment
    ✔ revert if FRAX address is 0
    ✔ revert if sFRAX address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if address is not valid sFrax address
    ✔ should get correct price of sFrax

SFrxETHOracle unit tests
  deployment
    ✔ revert if frxETH address is 0
    ✔ revert if sfrxETH address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if address is not valid sfrxETH address
    ✔ should get correct price of sfrxETH

SlisBNBOracle unit tests
  deployment
    ✔ revert if SynclubManager address is 0
    ✔ revert if slisBNB address is 0
    ✔ revert if resilientOracle address is 0
    ✔ should deploy contract
  getPrice
    ✔ revert if slisBNB address is wrong
    ✔ should get correct price

StkBNBOracle unit tests
```

```
    deployment
      ✔ revert if stakePool address is 0
      ✔ revert if stkBNB address is 0
      ✔ revert if resilientOracle address is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if ankrBNB address is wrong
      ✔ should get correct price

  WBETHOracle unit tests
    deployment
      ✔ revert if WBETH address is 0
      ✔ revert if ETH address is 0
      ✔ revert if resilientOracle address is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if WBETH address is wrong
      ✔ should get correct price

  WeETHOracle unit tests
    deployment
      ✔ revert if liquidity pool address is 0
      ✔ revert if weETH address is 0
      ✔ revert if eETH address is 0
      ✔ revert if resilient oracle address is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if address is not valid weETH address
      ✔ should get correct price of weETH

  WstETHOracle unit tests
    deployment
      ✔ revert if wstETH address is 0
      ✔ revert if stETH address is 0
      ✔ revert if ResilientOracle address is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if wstETH address is wrong
      ✔ should get correct price


  128 passing (9s)
```

Code Coverage

The branch coverage of the system stands at 92.16%, which means that the test suite is of a sufficient quality. It is always good to attempt to improve the branch coverage to 100%.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/oracles/	94.3	81	98.15	95.53	
AnkrBNBOracle.sol	100	100	100	100	
BNBxOracle.sol	100	100	100	100	
OneJumpOracle.sol	100	100	100	100	
PendleOracle.sol	100	75	100	100	
SFraxOracle.sol	100	100	100	100	
SFrxETHOracle.sol	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
SlisBNBOracle.sol	100	100	100	100	
StkBNBOracle.sol	100	50	100	83.33	45
WBETHOracle.sol	100	100	100	100	
WeETHOracle.sol	100	100	100	100	
WstETHOracle.sol	100	100	100	100	
contracts/oracles/common /	100	100	100	100	
CorrelatedTokenOracle.sol	100	100	100	100	
All files	99.52	92.16	99.85	98.24	

Changelog

- 2024-04-08 - Initial report
- 2024-04-10 - 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&aspo; 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 or 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 your 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 other 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 linked 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.

Quantstamp