Quantstamp

Venus - Oracle Update

Executive Summary

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

Type	Oracle
Timeline	2025-03-17 through 2025-03-25
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	None
Source Code	• VenusProtocol/oracle #1a14221
Auditors	• Rabib Islam Senior Auditing Engineer • Ibrahim Abouzied Auditing Engineer • Mostafa Yassin Auditing Engineer

Documentation quality	Medium
Test quality	Low
Total Findings	3 Fixed: 2 Acknowledged: 1
High severity findings	0
Medium severity findings	1 Acknowledged: 1
Low severity findings	2 Fixed: 2
Undetermined severity findings	0
Informational findings	0

Summary of Findings

The present audit concerns oracle contracts for Venus, a money market and stablecoin protocol. In particular, the `ResilientOracle` contract aims to provide prices from a main oracle while validating them against other oracles, and the `CorrelatedTokenOracle` aims to provide exchange rates that are limited in a particular fashion by an expected growth rate.

Of the issues found, we note in particular that when a pivot oracle is either not assigned or not enabled, the main oracle price is used without any validation, even leaving out the fallback oracle.

In addition, we present concerns regarding the capped oracle design as well as potential price staleness.

The test suite stands to be improved substantially in its current state. We suggest increasing branch coverage to over 90%.

We recommend all issues be addressed.

Update: VEN-1 was Acknowledged, and the remaining issues were Fixed. We continue to recommend improvement of the test suite.

ID	DESCRIPTION	SEVERITY	STATUS
VEN-1	Main Oracle Price Not Validated when Pivot Not Enabled	Medium	Acknowledged
VEN-2	Capped Oracle Design Assumes Uniform Growth	Low	Fixed
VEN-3	Potential Single-Transaction Price Staleness	Low	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

Files Included

- `contracts/ResilientOracle.sol`
- `contracts/interfaces/ICappedOracle.sol`
- `contracts/lib/Transient.sol`
- `contracts/oracles/common/CorrelatedTokenOracle.sol`
- `contracts/oracles/AnkrBNBOracle.sol`
- `contracts/oracles/BNBxOracle.sol`
- `contracts/oracles/ERC4626Oracle.sol`
- `contracts/oracles/EtherfiAccountantOracle.sol`
- `contracts/oracles/OneJumpOracle.sol`
- `contracts/oracles/PendleOracle.sol`
- `contracts/oracles/SFraxOracle.sol`
- `contracts/oracles/SlisBNBOracle.sol`
- `contracts/oracles/StkBNBOracle.sol`
- `contracts/oracles/WBETHOracle.sol`
- `contracts/oracles/WeETHAccountantOracle.sol`
- `contracts/oracles/WeETHOracle.sol`
- `contracts/oracles/WstETHOracleV2.sol`

Repo: `VenusProtocol/oracle/1a14221dc44d5b0a126baf05158c8946fd280195`

Operational Considerations

The project expects to rely on oracles such as Chainlink, Binance, onchain TWAP oracles, as well as other onchain data. To the extent these source projects are vulnerable to attack, the Venus oracle contracts are therefore vulnerable as well. However, measures are taken to reduce vulnerability, for example via price cross-validation.

Key Actors And Their Capabilities

The `ResilientOracle` contract is designed to be configured post-deployment, and as such it allows function-specific access control for its setter functions, as well as for pausing and unpausing that determines whether prices can be retrieved from the oracle.

Findings

VEN-1
Main Oracle Price Not Validated when Pivot Not Enabled

Medium | Acknowledged

Update

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

```
Intended functionality
  An alternative configuration we have for some assets is to set the same oracle for the PIVOT
and the FALLBACK roles. That way, the MAIN price will be compared to the “secondary” oracle,
and if the discrepancy is big enough, the price of the “secondary” market will be considered
```

File(s) affected: `ResilientOracle.sol`

Description: In addition to using a main oracle, typically Chainlink, for getting prices, the `ResilientOracle` contract can use a pivot oracle, typically a TWAP oracle, in order to validate the main oracle's price. However, if a price from the pivot oracle is not available, the main price can be validated by a fallback oracle, typically getting prices from Binance.

However, if the pivot oracle is not enabled, or if the pivot oracle is not set to a non-zero address, a pivot oracle is not used to validate the main oracle's price. Whereas it would make sense in this case to validate the main oracle's price using the fallback oracle, this is not done either. Instead, at `ResilientOracle.sol#L433`, we see that the main oracle's price is returned along with a flag that dubiously indicates that the main oracle's price has been validated by the pivot oracle, and hence, per `ResilientOracle.soL#L390`, the main oracle's price is returned without validating it against the fallback oracle. Hence, the main oracle's price is being returned without any validation whatsoever. In case the main oracle is compromised, this may lead to returning incorrect or inaccurate prices.

Recommendation: When the pivot oracle is not enabled, or when the pivot oracle is set to the zero address, consider still validating the main oracle's price against the fallback oracle's price in order to ensure that at least two oracles agree on the same price within a margin of error.

VEN-2 Capped Oracle Design Assumes Uniform Growth

Low | Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `bc2333903721c1dfe67e9202e19f7f862bfd0236`, `b5d7106c5e5e3844d4219dcb163920d4dcda9800`.
The client provided the following explanation:

```
• Review and evaluate whether the assumption of uniform growth applies to all correlated
  tokens.
    The annual growth rate should be set with a margin to “absorb” non-uniform growth. The
  Risk Manager will manage this
• Consider updating updateSnapshot() to use the fetched exchange rate.
    A new gap has been added to the minimum of the current exchange rate and the computed
  maximum, when automatically updating the snapshot. See
```

https://github.com/VenusProtocol/oracle/pull/239/commits/bc2333903721c1dfe67e9202e19f7f862bfd0236
- Consider implementing a function to allow for manually correcting an oracle if its caps become consistently restrictive.
  The following setters were added in https://github.com/VenusProtocol/oracle/pull/239/commits/b5d7106c5e5e3844d4219dcb163920d4dcda9800
- setSnapshot
- setGrowthRate
- setSnapshotGap

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

Description: The new capped oracle design calculates a maximum allowed exchange rate by converting a configured annual growth rate into a per-second growth rate and using a snapshot of a previous exchange rate to compute future maximum allowed rates, updating this snapshot periodically. `getPrice()` returns the minimum of the actual fetched exchange rate and the calculated maximum increase from the last snapshot.

This assumes that the maximum annualized growth rate is evenly distributed over every second of the year. However, if a correlated token experiences short-term growth spikes that temporarily exceed this uniform rate (balanced by slower growth during other periods) the oracle will cap the price during these high-growth intervals. This capping could lead to prices that do not accurately reflect the token's true market value during periods of rapid growth. This is compounded by the `updateSnapshot()` function also taking the minimum of the current exchange rate and the computed maximum, making prices even more conservative.

Recommendation:
- Review and evaluate whether the assumption of uniform growth applies to all correlated tokens.
- Consider updating `updateSnapshot()` to use the fetched exchange rate.
- Consider implementing a function to allow for manually correcting an oracle if its caps become consistently restrictive.

VEN-3 Potential Single-Transaction Price Staleness

Low Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `65c06ef4a668abe54cd09b7d08fdfe2b102226ac` .
The client described their changes as follows:

- caching can be enable/disable per asset
- exchange rates (that are read onchain) are not cached anymore
- remove of TWAP interactions (not related with Capped or Cached prices, but we haven't used ever the TWAP oracle and we prefer to remove that dead code)

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

Description: The `CorrelatedTokenOracle` and `ResilientOracle` now cache price updates for the duration of a transaction. If the transaction includes any interactions that would change the returned oracle prices after the initial read, the oracles will return stale cached values that may be exploited.

This would require the caller being able to cause a change in prices over the course of a single transaction. This is unlikely to occur if the oracle's main price comes from off-chain sources. However, if the price is derived on-chain data then it is more sensitive to on-chain trades and may open an arbitrage opportunity between the real and cached prices.

Recommendation: For each asset, consider whether the price can become stale within a single transaction. If the price source is calculated on-chain and can be meaningfully impacted by the caller in a single transaction, consider doing a risk assessment and disabling caching for such assets

Auditor Suggestions

S1 Clean Code

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `d17516bd367a811a635d7aaa38a8e4404b7185a7`, `58dbffb1ce350b47d13a30ae87c5d97daba6a35c`.
The client provided the following explanation:

```
Remove unchecked:
https://github.com/VenusProtocol/oracle/pull/239/commits/d17516bd367a811a635d7aaa38a8e4404b7185a7
  Remove unused var:
https://github.com/VenusProtocol/oracle/pull/239/commits/58dbffb1ce350b47d13a30ae87c5d97daba6a35c
```

Description: Certain changes can be made in order to make the code more succinct.

1. Remove `unchecked { ++i }` as it was deprecated in Solidity 0.8.22.
2. Remove the unused `address asset` from `CorrelatedTokenOracle._calculatePrice()`.

Recommendation: Make the above-suggested changes.

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.

Test Suite Results

Tests about updating snapshots and asset prices should be implemented.

Update: Two minor tests were added for `ERC4626Oracle`.

```
AnkrBNBOracle unit tests
  deployment
    ✔ revert if ankrBNB address is 0 (43ms)
    ✔ 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
  ✔ fetch BNB price
  ✔ price expired (55ms)
  ✔ set WBETH price
  ✔ fetch WBETH price
  ✔ revert when setting feed registry address and sid already available
  ✔ revert when feed registry address is zero (68ms)
  ✔ fetch price from direct feed registry

bound validator
  add validation config
    ✔ length check
    ✔ validation config check
    ✔ config added successfully & event check
  validate price
    ✔ validate price (176ms)

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

ERC4626Oracle unit tests
  deployment
    ✔ revert if FRAX address is 0
    ✔ revert if sFRAX address is 0
    ✔ should deploy contract (84ms)
  getPrice
    ✔ revert if address is not valid sFrax address
    ✔ should get correct price of sFrax
    ✔ cache exchange rate in transient storage (66ms)

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

  PendleOracle unit tests
    deployment
      ✔ revert if market address is 0
      ✔ revert if ptOracle address is 0
      ✔ revert if PT token address is 0
      ✔ revert if underlying token address is 0
      ✔ revert if ResilientOracle address is 0
      ✔ revert if TWAP duration is 0
      ✔ revert if invalid TWAP duration
    getPrice
      ✔ revert if getPrice argument is not the configured PT token (141ms)
      ✔ should get correct price for PT_TO_ASSET rate kind (150ms)
      ✔ should get correct price for PT_TO_SY rate kind (157ms)
      ✔ should adjust for underlying decimals (157ms)

  ReferenceOracle
    setOracle
      ✔ reverts if the asset is zero address
      ✔ reverts if called by a non-admin
      ✔ emits OracleConfigured event
      ✔ sets the new oracle
      ✔ can unset the oracle
    getPrice
      ✔ returns resilient oracle price if there's no oracle configured
      ✔ returns the other oracle price if there's an oracle set for asset
    getPriceAssuming
      ✔ returns the assumed price if no oracle is configured
      ✔ returns the assumed price even if the oracle is configured

  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 (144ms)
        ✔ token config added successfully & events check (144ms)
      batch add token configs
        ✔ length check
        ✔ token config added successfully & data check (373ms)
    change oracle
      set oracle
        ✔ null check (154ms)
        ✔ existance check (110ms)
        ✔ oracle set successfully & data check (151ms)
    get underlying price
      ✔ revert when protocol paused (39ms)
      ✔ 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 (52ms)
      ✔ disable pivot oracle (40ms)
      ✔ enable fallback oracle (148ms)
      ✔ Return fallback price when fallback price is validated successfully with pivot oracle
(39ms)
      ✔ 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 SfrxEthFraxOracle address is 0
      ✔ revert if sfrxETH address is 0
      ✔ revert if price different is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if address is not valid sfrxETH address
      ✔ revert if price difference is more than allowed
      ✔ should get correct price of sfrxETH

  SequencerChainlinkOracle
    ✔ Should revert if sequencer is down
    ✔ Should revert if sequencer is up, but GRACE_PERIOD has not passed
    ✔ Should return price

  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

  WstETHOracleV2 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

  ZkETHOracle
    deployment
      ✔ revert if WETH address is 0
      ✔ revert if zkETH address is 0
      ✔ should deploy contract
    getPrice
      ✔ revert if address is not valid zkETH address
      ✔ should get correct price of zkETH

  CorrelatedTokenOracle
    Max Price Logic
      ✔ should return the correct price capped by the max allowed price


  144 passing (20s)
```

Code Coverage

Branch coverage for `ResilientOracle` is 75.86%, and that of `CorrelatedTokenOracle` is around 43.33%, while it is 100% for `Transient.sol`. Oracle contracts derived from `CorrelatedTokenOracle` generally have quite high branch coverage. We recommend improving branch coverage for all in-scope contracts to above 90%.

Update: Branch coverage `CorrelatedTokenOracle` has fallen to 36.67%.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	81	77.27	86.21	83.04	
ReferenceOracle.sol	100	87.5	100	100	
ResilientOracle.sol	76.54	75.86	80.95	78.65	... 443,472,488
contracts/interfaces/	100	100	100	100	
FeedRegistryInterface.sol	100	100	100	100	
IAccountant.sol	100	100	100	100	
IAnkrBNB.sol	100	100	100	100	
ICappedOracle.sol	100	100	100	100	
IERC4626.sol	100	100	100	100	
IEtherFiLiquidityPool.sol	100	100	100	100	
IPStakePool.sol	100	100	100	100	
IPendlePtOracle.sol	100	100	100	100	
ISFrax.sol	100	100	100	100	
ISfrxEthFraxOracle.sol	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
IStETH.sol	100	100	100	100	
IStaderStakeManager.sol	100	100	100	100	
ISynclubStakeManager.sol	100	100	100	100	
IWBETH.sol	100	100	100	100	
IZkETH.sol	100	100	100	100	
OracleInterface.sol	100	100	100	100	
PublicResolverInterface.sol	100	100	100	100	
SIDRegistryInterface.sol	100	100	100	100	
VBep20Interface.sol	100	100	100	100	
contracts/lib/	50	100	50	50	
Transient.sol	50	100	50	50	11,12
contracts/oracles/	88.95	79.59	90.16	88.73	
AnkrBNBOracle.sol	100	100	100	100	
BNBxOracle.sol	100	100	100	100	
BinanceOracle.sol	87.8	71.43	100	95.74	130,131
BoundValidator.sol	96.15	83.33	100	95.65	115
ChainlinkOracle.sol	100	92.31	100	100	
ERC4626Oracle.sol	100	100	100	100	
EtherfiAccountantOracle.sol	0	100	0	0	39,40,48
OneJumpOracle.sol	100	100	100	100	
PendleOracle.sol	100	83.33	100	100	
SFraxOracle.sol	100	100	100	100	
SFrxETHOracle.sol	100	75	100	100	
SequencerChainlinkOracle.sol	100	87.5	100	100	
SlisBNBOracle.sol	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
StkBNBOracle.sol	100	50	100	83.33	57
WBETHOracle.sol	100	100	100	100	
WeETHAccountantOracle.sol	0	100	0	0	38,39,47
WeETHOracle.sol	100	100	100	100	
WstETHOracle.sol	0	0	0	0	... 64,67,70,73
WstETHOracleV2.sol	100	100	100	100	
ZkETHOracle.sol	100	100	100	100	
contracts/oracles/common/	62.16	43.33	83.33	63.83	
CorrelatedTokenOracle.sol	62.16	43.33	83.33	63.83	... 132,133,148
All files	82.96	73.2	87.76	83.38	

Changelog

- 2025-03-25 - Initial report
- 2025-04-23 - 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

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.