FAIRYPROOF

Venus

AUDIT REPORT

Version 1.0.0

Serial No. 2021111600012017

Presented by Fairyproof

November 16, 2021

01. Introduction

This document includes the results of the audit performed by the Fairyproof team on the Venus project.

Audit Start Time:

November 8, 2021

Audit End Time:

November 15, 2021

Audited Code's Github Repository:

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

Audited Code's Github Commit Number When Audit Started:

9a7718896faeac1cb1b7786a14525be4666081b8

Audited Code's Github Commit Number When Audit Ended:

9a7718896faeac1cb1b7786a14525be4666081b8

Audited Source Files:

The calculated SHA-256 values for the audited files when the audit was done are as follows:

```
AggregatorV2V3Interface.sol:
0xf54f779f19b2a09e56c90b463d9c94f67b38277b8f9677aaf697f2f23d1dd0a7

BEP20Interface.sol:
0x05bdba8f52f4b63a180f9ceb1d9d59bb8da2ab56179144639382f659e5d82b71

CarefulMath.sol:
0x0ae89f428a3f3ea509da787b5662945d9ce58432c61cd4076899410b18e2d0bc

Comptroller.sol:
0xd32a5d4d571f37057720a697c44a3d4d015a3ae7ccc9317db79140180e0ac24a

ComptrollerG1.sol:
0xe74886e5890821062da439f76ab6d985d38347483ee454a96845457f18913a04

ComptrollerG2.sol:
0x750aa0384d055b4736d607d9110ec2407ce7f1ec2bc1e8f7a9edc8bb7c8ac715

ComptrollerG3.sol:
0xb06909396cacc5da51bbaedc4da9b596800ece7b07621215750e081bee4edd33
```

```
ComptrollerG4.sol:
0xb0634825073593d881fd3236a1116ac00eade0eea7e40fad8602884a3d21dcfc

ComptrollerInterface.sol:
0x2ab3272aa3027db3c134e8f9228fa7ed09a571485db320228b5b752971238ef5

ComptrollerStorage.sol:
0x9c5f9cef7b3fa9897089154e8d9d2a9594a817dd00d598ed674b6895787ab2c6

Context.sol:
0x404217d9bcfd69dc51adee1870927b9456c1cb8ce944f6bcf9e198f3eabf2745

DAIInterestRateModelV2.sol:
0x24629b17e86f367bf2428cfab5e3706f00c67675d9942a30437c919debf69893

EIP20Interface.sol:
0x63947892fbba221de8eefdcca582f92749412816c4f0ff36561c5e9166d7073e

EIP20NonStandardInterface.sol:
0xd2db1e7446818cbae8ae2446145d0b51f2189cc290e1d5384cc8c4070150cddf

ErrorReporter.sol:
0x91e0013bc9f1092402f2e3c1dd80bbd982a6fce1b48d3c2aeca2556d25315ba0

Exponential.sol:
0x0fd7233bb15d1eb6a5e398ce391ca48b96a0fb8e9b15a4c66d5912086da233b1

ExponentialNoError.sol:
0x418ae000ba621eb3e8ef0e4f2347310f0c2e5f3bb75b183681d8bf67c7c14b11

GovernorAlpha.sol:
0x5cca72419caa350262d564d9643a1a05a6caa2538515c9f35c2cc0f06a48842f

GovernorAlpha2.sol:
0xd1ef4d430cb64e60d186f31e087022efb64ccfa6172d78e5ab115bc560dab012

SXP.sol:
0x911a44ff570fe99a584a9e11401c59d521997eb005228c174a57567e8cd6e431

XVS.sol:
0xd925d89b6bf4b2c7b6c15bcc97d8d785b394c7177bb9cbe5e8cbb746d70b9f05

InterestRateModel.sol:
0xa5e9ccea5986e95840599c28f568d38e7c5bfd8fbf1ea90f1f961f6003ce5204

JumpRateModel.sol:
0xe10d6dcf1e3b1a056ca90c4c7ea69b18562c50cd44f6eccca47590b73f985831
```

```
VenusLens.sol:
0x678a7079ef7d01eb6ddd382940f5c612306cf80bf5a516ca077120201e757c66

Maximillion.sol:
0x06ab97600361b1cd915d3a56295ee8c4bacd86d062e57c262332862670b2ed06

Ownable.sol:
0xecb7884d2448a29af4d3751b6c2a0a4c700522d834962f6c45e66ebca5392703

PriceOracle.sol:
0x50f9d59d577fb476267fa3f93f6111744de046d4a586263987bac96b2993d5e6

PriceOracleProxy.sol:
0xd1ee660ff186cead9284e708421b7468d9f72076bee399f1dae4723450f50a5e

Reservoir.sol:
0x5ff6630cc50b357af57b0990feffea771a638d21360ab9a2fea2de095cc9a23c

SafeMath.sol:
0x358b9dcc6321bc1feba5a5e0b5353aeb12384401b80f51613175560fa5259ab5

SimplePriceOracle.sol:
0xf8c2b9458caaa42bd31a8dd7213c29992f723bb00c5d2f3edd739c101f44ef02

Timelock.sol:
0xea4204fc8c5c72a5f4984177c209a16be5d538f1a3ee826744c901c21d27e382

Unitroller.sol:
0x6b7011169a46cb1e5ef07df07f0d27d5e9a506458c9ba9e89bce69535c0a42d0

Address.sol:
0x7b3c22c73101d02202123e7a3568e26c2a0f10b5048c23ec22268c371843bbf7

IBEP20.sol:
0x05d1b23fa6c9443ae7b25b87dd339a1f75f5eaee5dfe7da1039bc8e07bbbf528

SafeBEP20.sol:
0xb6f56620b49975b338e0f5516ca037a689ff3f93caa4951d78f5eb1bf87a0cbd

SafeMath.sol:
0xccbc65eddc0fe23db1360af754dfc534f2ab28ab1d2e79c1ca0cc9420a96dc58

VAI.sol:
0xe651466f024bece0a9dfc8c2bf8bb8711ffcca44a4515b520765db8f146228f2

lib.sol:
0xf86eb92bd80b9f3a9cbcdc3572ca167d51b31ed381fdcbdfb91516d55b62cf3e

VAIController.sol:
```

```
0xa24d2d8d0afdedaac0689b18152459abce09f87ee7b582c2af94f28cfc91f1ff

VAIControllerG1.sol:
0x4fa710d7f30c098e933775daa0e6d451ba916e5cb635ae03e16547dd21217790

VAIControllerInterface.sol:
0x60c259a6468838e5620d2ad89d809a451786873f15fda78dc29b4b1429dbea67

VAIControllerStorage.sol:
0xe53b2d111399b4f3b3fdc96f09d2bb9aaa1ab964d5f4a799afb90e0d200d93df

VAIUnitroller.sol:
0x46a30e7c27812c64bf2154aabd1377dc6ff96637e7520b654bc2e09c39cff450

VBNB.sol:
0x28a77a82e5c96dce259394454b5a7cdd234dd4f80a36af02aa03783e292b0106

VBep20.sol:
0xb2b2a34d667b27d694f7c89210edd249612fa266c9c847589e1094156f10c11d

VBep20Delegate.sol:
0x2974210a3a194160f288e17a0012bef948001844787eeacc7f74a62f995d84f6

VBep20Delegator.sol:
0x2d4e7f9898fccdd2cba8b8266114254cd38eb087ff5a0386ed1f99c9dcfc5994

VBep20Immutable.sol:
0x6e4cb4a0ec11fc97ddc719e901211f6bffef4fd58964dac15380c208baf37e79

VDaiDelegate.sol:
0x455623a33adc44a024669315d57ccf3f7b75b5d6852145405764606 7d02ebf53

VToken.sol:
0x6cf8743917662e2b8216510611f583c1ed921ea3c36de3bb52ab719586643fff

VTokenInterfaces.sol:
0xd8fcb7dd5819eb176e304808a4c634e5dd80dcc1c07370c5c3671850b8d9af01

VTreasury.sol:
0x9bb73749d520c13d60226e3db0c95e639269f330d1d6313f477c4ffece1f9af2

VXvsLikeDelegate.sol:
0x03af6e5c559dbd42d2d90a62befb67a34b6c8500c218d8e90b52977d35141a27

VAIVault.sol:
0x5b9775c0af201dfa077a3ce1d398716a3accfcce47f11a8141bd04f52710f873

VAIVaultErrorReporter.sol:
0xff1e405eeacf2cd9f0286eaaf68c855ab7b0e9935326ac5fe259d827c39354ea
```

```

VAIVaultProxy.sol:
0xe9c2f153331da1fdf5d0fd0bcf4b89ae3a6007691b988e7042c05499ef9348ff

VAIVaultStorage.sol:
0xa7fe5ca088deced822f661545e3f6242e2c741a5cb494b2e6a0dedfa5d2369ea

VenusChainlinkOracle.sol:
0x53d039e1bf5a5ea28250739937fb9ff1218191d51a43f5786acb7eac052f478f

VenusPriceOracle.sol:
0xf6e90b073963ed8a98659ce51b1c43f557c1cebdf24713111813343bdcd26e7f

WhitePaperInterestRateModel.sol:
0x92413b26baefc323326a8e1936d6f0ec62626800d80e98e04303127da3302006
```

The source files audited include all the files with the extension "sol" as follows:

```
contracts/
├── AggregatorV2V3Interface.sol
├── BEP20Interface.sol
├── CarefulMath.sol
├── Comptroller.sol
├── ComptrollerG1.sol
├── ComptrollerG2.sol
├── ComptrollerG3.sol
├── ComptrollerG4.sol
├── ComptrollerInterface.sol
├── ComptrollerStorage.sol
├── Context.sol
├── DAIInterestRateModelV2.sol
├── EIP20Interface.sol
├── EIP20NonStandardInterface.sol
├── ErrorReporter.sol
├── Exponential.sol
├── ExponentialNoError.sol
├── Governance
│   ├── GovernorAlpha.sol
│   ├── GovernorAlpha2.sol
│   ├── SXP.sol
│   └── XVS.sol
├── InterestRateModel.sol
├── JumpRateModel.sol
├── Lens
│   └── VenusLens.sol
├── Maximillion.sol
├── Ownable.sol
├── PriceOracle.sol
├── PriceOracleProxy.sol
├── Reservoir.sol
├── SafeMath.sol
├── SimplePriceOracle.sol
├── Timelock.sol
├── Unitroller.sol
├── Utils
│   ├── Address.sol
│   ├── IBEP20.sol
│   ├── SafeBEP20.sol
│   └── SafeMath.sol
├── VAI
│   ├── VAI.sol
│   └── lib.sol
├── VAIController.sol
├── VAIControllerG1.sol
├── VAIControllerInterface.sol
├── VAIControllerStorage.sol
├── VAIUnitroller.sol
├── VBNB.sol
├── VBep20.sol
├── VBep20Delegate.sol
├── VBep20Delegator.sol
├── VBep20Immutable.sol
├── VDaiDelegate.sol
├── VToken.sol
├── VTokenInterfaces.sol
├── VTreasury.sol
├── VXvsLikeDelegate.sol
├── Vault
│   ├── VAIVault.sol
│   ├── VAIVaultErrorReporter.sol
│   ├── VAIVaultProxy.sol
│   └── VAIVaultStorage.sol
├── VenusChainlinkOracle.sol
├── VenusPriceOracle.sol
└── WhitePaperInterestRateModel.sol
```

The goal of this audit is to review Venus' solidity implementation for its decentralized lending application, study potential security vulnerabilities, its general design and architecture, and uncover bugs that could compromise the software in production.

We make observations on specific areas of the code that present concrete problems, as well as general observations that traverse the entire codebase horizontally, which could improve its quality as a whole.

This audit only applies to the specified code, software or any materials supplied by the Venus team for specified versions. Whenever the code, software, materials, settings, enviroment etc is changed, the comments of this audit will no longer apply.

— Disclaimer

Note that as of the date of publishing, the contents of this report reflect the current understanding of known security patterns and state of the art regarding system security. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk.

The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from offchain sources are not extended by this review either.

Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit is recommended after the issues covered are fixed.

To the fullest extent permitted by law, we disclaim 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.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, 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, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

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.

— Methodology

The above files' code was studied in detail in order to acquire a clear impression of how the its specifications were implemented. The codebase was then subject to deep analysis and scrutiny, resulting in a series of observations. The problems and their potential solutions are discussed in this document and, whenever possible, we identify common sources for such problems and comment on them as well.

The Fairyproof auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Fairyproof to make sure we understand the size, scope, and functionality of the project's source code.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.

iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Fairyproof describe.

2. Testing and automated analysis that includes the following:

   i. 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 the test cases.

   ii. 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 source code to improve maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

— Structure of the document

This report contains a list of issues and comments on all the above source files. Each issue is assigned a severity level based on the potential impact of the issue and recommendations to fix it, if applicable. For ease of navigation, an index by topic and another by severity are both provided at the beginning of the report.

— Documentation

For this audit, we used the following sources of truth about how the decentralized lending application should work:

https://venus.io/

whitepaper

These were considered the specification.

— Comments from Auditor

Serial Number	Auditor	Audit Time	Result
2021111600012017	Fairyproof Security Team	November 8, 2021 - November 15, 2021	Low Risk

Summary:

The Fairyproof security team used its auto analysis tools and manual work to audit the project. During the audit two risks of low-severity were discovered.

02. About Fairyproof

Fairyproof is a leading technology firm in the blockchain industry, providing consulting and security audits for organizations. Fairyproof has developed industry security standards for designing and deploying blockchain applications.

03. Introduction to Venus

Venus Protocol ("Venus") is an algorithmic-based money market system designed to bring a complete decentralized finance-based lending and credit system onto Binance Smart Chain. Venus enables users to utilize their cryptocurrencies by supplying collateral to the network that may be borrowed by pledging over-collateralized cryptocurrencies. This creates a secure lending environment where the lender receives a compounded interest rate annually (APY) paid per block, while the borrower pays interest on the cryptocurrency borrowed. These interest rates are set by the protocol in a curve yield, where the rates are automated based on the demand of the specific market, such as Bitcoin. The difference of Venus from other money market protocols is the ability to use the collateral supplied to the market not only to borrow other assets but also to mint synthetic stablecoins with over-collateralized positions that protect the protocol. These synthetic stablecoins are not backed by a basket of fiat currencies but by a basket of cryptocurrencies. Venus utilizes the Binance Smart chain for fast, low-cost transactions while accessing a deep network of wrapped tokens and liquidity.

04. Major functions of audited code

The audited code implements the following functions:

- Users deposit crypto assets to earn interests;
- When the price of a deposited asset is lower than the price of the borrowed asset, liquidation of the deposited asset will be triggered.

Note: the VToken token is the certificate token for a deposited asset. Users who deposit assets will get VToken tokens and should keep them safe and secure.

05. Coverage of issues

The issues that the Fairyproof team covered when conducting the audit include but are not limited to the following ones:

- Re-entrancy Attack
- Replay Attack
- Reordering Attack
- DDos Attack
- Transaction Ordering Attack
- Race Condition
- Access Control
- Integer Overflow/Underflow
- Timestamp Attack
- Gas Consumption
- Inappropriate Callback Function
- Function Visibility
- Implementation Vulnerability
- Uninitialized Storage Pointer
- Arithmetic Precision
- Tx.origin
- Fake Deposit
- Shadow Variable
- Design Vulnerability
- Token Issurance
- Admin Rights
- Inappropriate Proxy Design
- Inappropriate Use of Slots
- Asset Security
- Contract Upgrade/Migration
- Code Improvement

06. Severity level reference

Every issue in this report was assigned a severity level from the following:

Critical severity issues need to be fixed as soon as possible.

High severity issues will probably bring problems and should be fixed.

Medium severity issues could potentially bring problems and should eventually be fixed.

Low severity issues are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

Neutral is not an issue or risk but a suggestion for code improvement.

07. List of issues by severity

Index	Description	Issue/Risk	Severity	Status
N1	Inappropriate Setting	Implementation Vulnerability	Low	N/A
N2	Cross-chain Replay Attack	Replay Attack	Low	N/A

08. Issue descriptions

[N1] [Low] Inappropriate Setting

Risk Severity: Low

Issue/Risk: Implementation Vulnerability

Description:

The setting of `totalReservesNew` in the `_addReservesFresh` function defined in line 1354 of the `VToken.sol` file is inappropriate. If the value of `totalReservesNew` is greater than `getCashPrior() + totalBorrows`, overflow will happen in line 56 of the `WhitePaperInterestRateModel.sol` file.

Recommendation:

Consider adding the following directive in line 1382 of the `VToken.sol` file.

```
require(totalReservesNew < getCashPrior() + totalBorrows, "add reserves unexpected
overflow");
```

Status: N/A

[N2] [Low] Cross-chain Replay Attack

Risk Severity: Low

Issue/Risk: Replay Attack

Description:

The `permit` function in line 120 of the `VAI.sol` file doesn't check the `chainId` therefore it may be exposed to cross-chain replay attacks.

Recommendation:

Consider reimplementing the code as follows:

```
//add a chainId variable
uint256 public chainId;

constructor(uint256 chainId_) public {
  //save this chainId
  chainId = chainId_;
}
//check the chainId in the permit function
function permit() {
    uint256 contextChainId;
    assembly {
        contextChainId := chainid()
    }
    require(chainId == contextChainId, "chainIds do not match");
}
```

Status: N/A

09. Recommendations to enhance the overall security

We list some recommendations in this section. They are not mandatory but will enhance the overall security of the system if they are adopted.

- N/A

FAIRYPROOF

https://medium.com/@FairyproofT

https://twitter.com/FairyproofT

https://www.linkedin.com/company/fairyproof-tech

https://t.me/Fairyproof_tech

Reddit: https://www.reddit.com/user/FairyproofTech