



**SOLID**Proof  
*Bring trust into your projects*

**Blockchain Security | Smart Contract Audits | KYC  
Development | Marketing**

MADE IN GERMANY

# Vitreus Audit

**Security Assessment  
30. August, 2023**

**For**



VITREUS



**SolidProof\_io**



**@solidproof\_io**

Disclaimer	3
Description	5
Project Engagement	5
Logo	5
Contract Link	5
Methodology	7
Used Code from other Frameworks/Smart Contracts (direct imports)	8
Tested Contract Files	9
Source Lines	10
Risk Level	10
Capabilities	11
Inheritance Graph	12
CallGraph	13
Scope of Work/Verify Claims	14
Modifiers and public functions	23
Source Units in Scope	25
Critical issues	26
High issues	26
Medium issues	26
Low issues	26
Informational issues	27
Audit Comments	28
SWC Attacks	29

# Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Uniswap, Uniswap, PancakeSwap etc’...)

**SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.**

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	14. January 2023	<ul style="list-style-type: none"><li>• Layout project</li><li>• Automated- /Manual-Security Testing</li><li>• Summary</li></ul>
v1.1	17. June 2023	<ul style="list-style-type: none"><li>• Reaudit</li></ul>
v1.2	30. August 2023	<ul style="list-style-type: none"><li>• Added Functionality Audit</li></ul>

## **Network**

Ethereum (ERC20)

Binance Smart Chain (BEP20)

## **Website**

<http://vitreus.io>

## **Telegram**

<https://t.me/VitreusChain>

## **Twitter**

Vitreus: <https://twitter.com/VitreusChain>

Chad, Founder: <https://twitter.com/CollabChad>

Brent, Co-Founder: <https://twitter.com/crypfoo12>

Baran: [https://twitter.com/Wayne\\_Lambeau](https://twitter.com/Wayne_Lambeau)

Jaren: <https://twitter.com/mrscryptorabbit>

Taylor: <https://twitter.com/tcrypto1199>

## **Facebook**

<https://www.facebook.com/VitreusChain>

## **Instagram**

<https://instagram.com/vitreuschain>

## **Discord**

<https://discord.gg/vitreus>

## **Youtube**

<https://www.youtube.com/@VitreusChain>

## **LinkedIn**

<https://www.linkedin.com/company/vitreus-chain>

## Description

A permissioned financial services distributed ledger from the infrastructure and software solutions company, Collaborative Digital.

## Project Engagement

During the 12th of January 2023, **Vitreus Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

## Logo



VITREUS

## Contract Link v1.2

- <https://github.com/Vitreus-Essentials/vitreus-contracts/blob/feature/affiliate-rewards/contracts/VitreusVoucher.sol>
- Commit: 75b942e

# Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
<b>Critical</b>	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
<b>High</b>	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
<b>Medium</b>	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
<b>Low</b>	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
<b>Informational</b>	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

# Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

## **Methodology**

The 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 SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
  - 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 SolidProof 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 those test cases.
  - ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

## Used Code from other Frameworks/Smart Contracts (direct imports)

Imported packages:

Dependency / Import Path	Count
@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	1
@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC721/IERC721Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC721/extensions/IERC721MetadataUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/cryptography/ECDSAUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol	1
@openzeppelin/contracts/token/ERC20/IERC20.sol	1
@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol	1
@openzeppelin/contracts/utils/introspection/ERC165Checker.sol	1
@uniswap/v2-core/contracts/interfaces/IUniswapV2Pair.sol	1
@uniswap/v2-periphery/contracts/interfaces/IUniswapV2Router02.sol	1
operator-filter-registry/src/upgradeable/DefaultOperatorFiltererUpgradeable.sol	1



## Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

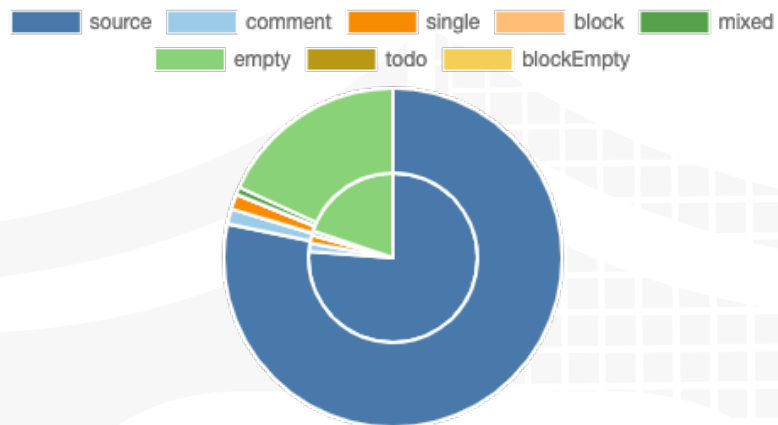
*A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.*

### v1.2

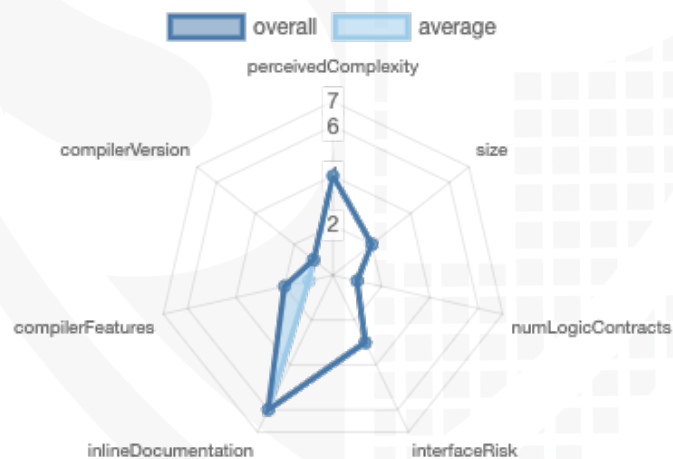
File Name	SHA-1 Hash
contracts/VitreusVoucher.sol	58cc34a8f549e484043fd695e1cdf76b4b75024c

# Metrics

## Source Lines v1.2



## Risk Level v1.2



## Capabilities

### Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	1	0	0	0

### Exposed Functions

*This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.*

Version	Public	Payable
1.0	44	5

Version	External	Internal	Private	Pure	View
1.0	37	52	1	2	33

### State Variables

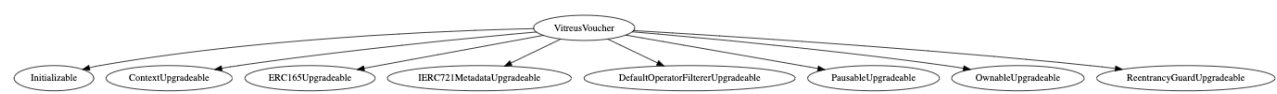
Version	Total	Public
1.0	17	1

### Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	0.8.18		yes	yes (1 asm blocks)	

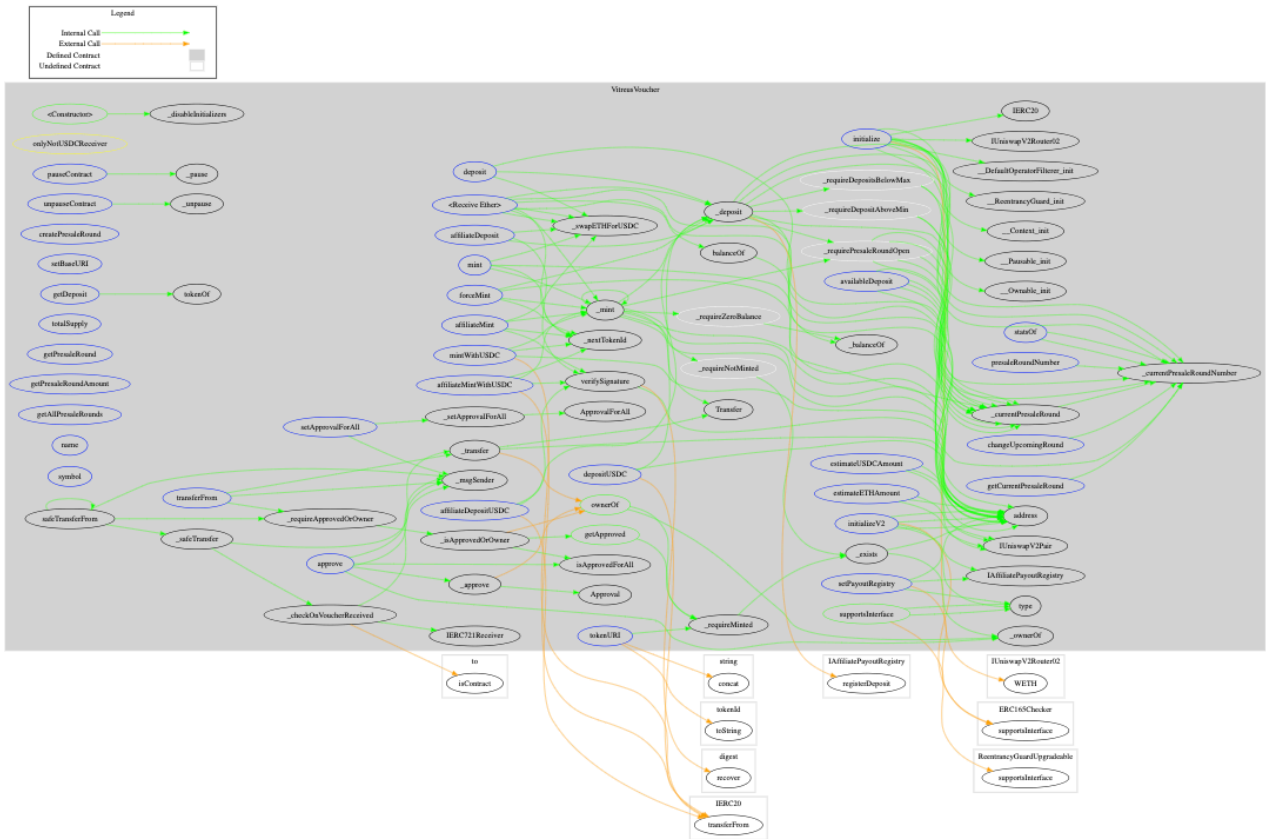
# Inheritance Graph

## v1.2



# CallGraph

## v1.2



## Scope of Work/Verify Claims

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1. Is contract an upgradeable
2. Correct implementation of Token standard
3. Deployer cannot mint any new tokens
4. Deployer cannot burn or lock user funds
5. Deployer cannot pause the contract
6. Deployer cannot set fees
7. Deployer cannot blacklist/antisnipe addresses
8. Overall checkup (Smart Contract Security)

## Is contract an upgradeable

Name	
Is contract an upgradeable?	Yes

Comments:

### v1.0

- Owner can deploy a new version of the contract which can change any limit and give owner new privileges
  - Be aware of this and do your own research for the contract which is the contract pointing to

## Correct implementation of Token standard

ERC721				
Function	Description	Exist	Tested	Verified
BalanceOf	Count all NFTs assigned to an owner	✓	✓	✓
OwnerOf	Find the owner of an NFT	✓	✓	✓
SafeTransferFrom	Transfers the ownership of an NFT from one address to another address	✓	✓	✓
SafeTransferFrom	See above - Difference is that this function has an extra data parameter	✓	✓	✓
TransferFrom	Transfer ownership of an NFT	✓	✓	✓
Approve	Change or reaffirm the approved address for an NFT	✓	✓	✓
SetApprovalForAll	Enable or disable approval for a third party ("operator") to manage all of `msg.sender`'s assets	✓	✓	✓
GetApproved	Get the approved address for a single NFT	✓	✓	✓
IsApprovedForAll	Query if an address is an authorized operator for another address	✓	✓	✓
SupportsInterface	Query if a contract implements an interface	✓	✓	✓
Name	Provides information about the name	✓	✓	✓
Symbol	Provides information about the symbol	✓	✓	✓
TokenURI	Provides information about the TokenUri	✓	✓	✓



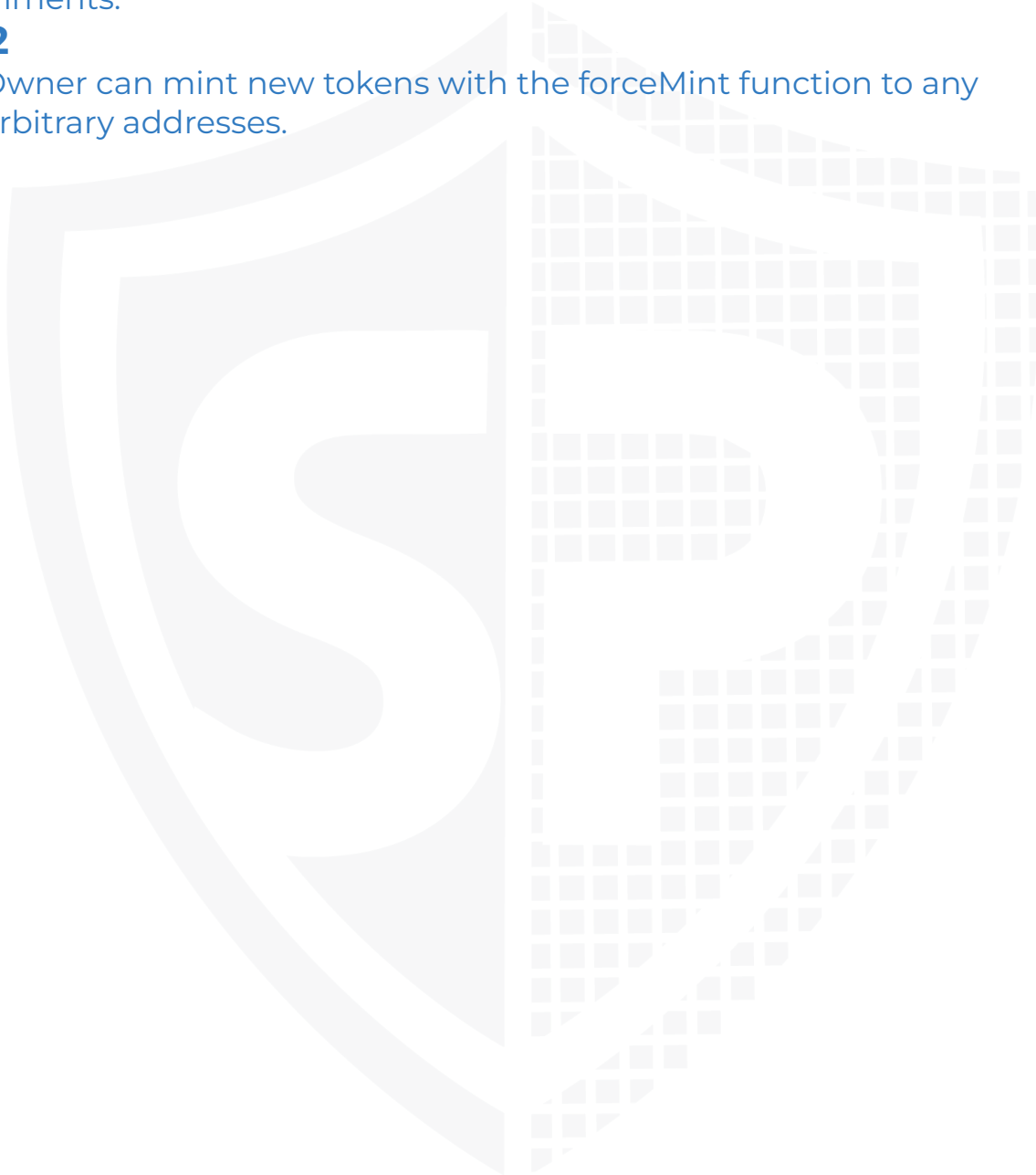
## Deployer cannot mint any new tokens

Name	Exist	Tested	Status
Deployer cannot mint	✓	✓	✗

Comments:

### v1.2

- Owner can mint new tokens with the forceMint function to any arbitrary addresses.



## Deployer cannot burn or lock user funds

Name	Exist	Tested	Status
Deployer cannot lock	—	—	—
Deployer cannot burn	—	—	—



## Deployer cannot pause the contract

Name	Exist	Tested	Status
Deployer cannot pause	✓	✓	✗

Comments:

### v1.2

- The owner can pause the mintWithUSDC and depositUSDC function. But keep in mind that the contract is an upgradeable contract that means that the deployer is able to add new functionalities to the contract.



## Deployer cannot set fees

Name	Exist	Tested	Status
Deployer cannot set fees over 25%	—	—	—
Deployer cannot set fees to nearly 100% or to 100%	—	—	—



## Deployer can blacklist/antisnipe addresses

Name	Exist	Tested	Status
Deployer cannot blacklist/antisnipe addresses	—	—	—



## Overall checkup (Smart Contract Security)


Tested	Verified
✓	✓

### Legend

Attribute	Symbol
Verified / Checked	✓
Partly Verified	🚩
Unverified / Not checked	✗
Not available	—

# Modifiers and public functions

## v1.2



```
initialize
initializer
initializeV2
reinitializer
mint
whenNotPaused
onlyNotUSDCReceiver
affiliateMint
whenNotPaused
onlyNotUSDCReceiver
deposit
whenNotPaused
onlyNotUSDCReceiver
affiliateDeposit
whenNotPaused
onlyNotUSDCReceiver
mintWithUSDC
whenNotPaused
onlyNotUSDCReceiver
affiliateMintWithUSDC
whenNotPaused
onlyNotUSDCReceiver
depositUSDC
whenNotPaused
onlyNotUSDCReceiver
affiliateDepositUSDC
whenNotPaused
onlyNotUSDCReceiver
pauseContract
onlyOwner
unpauseContract
onlyOwner
createPresaleRound
onlyOwner
forceMint
onlyOwner
changeUpcomingRound
onlyOwner
setBaseURI
onlyOwner
setPayoutRegistry
onlyOwner
approve
onlyAllowedOperatorApproval
setApprovalForAll
onlyAllowedOperatorApproval
transferFrom
onlyAllowedOperator
safeTransferFrom
onlyAllowedOperator
```

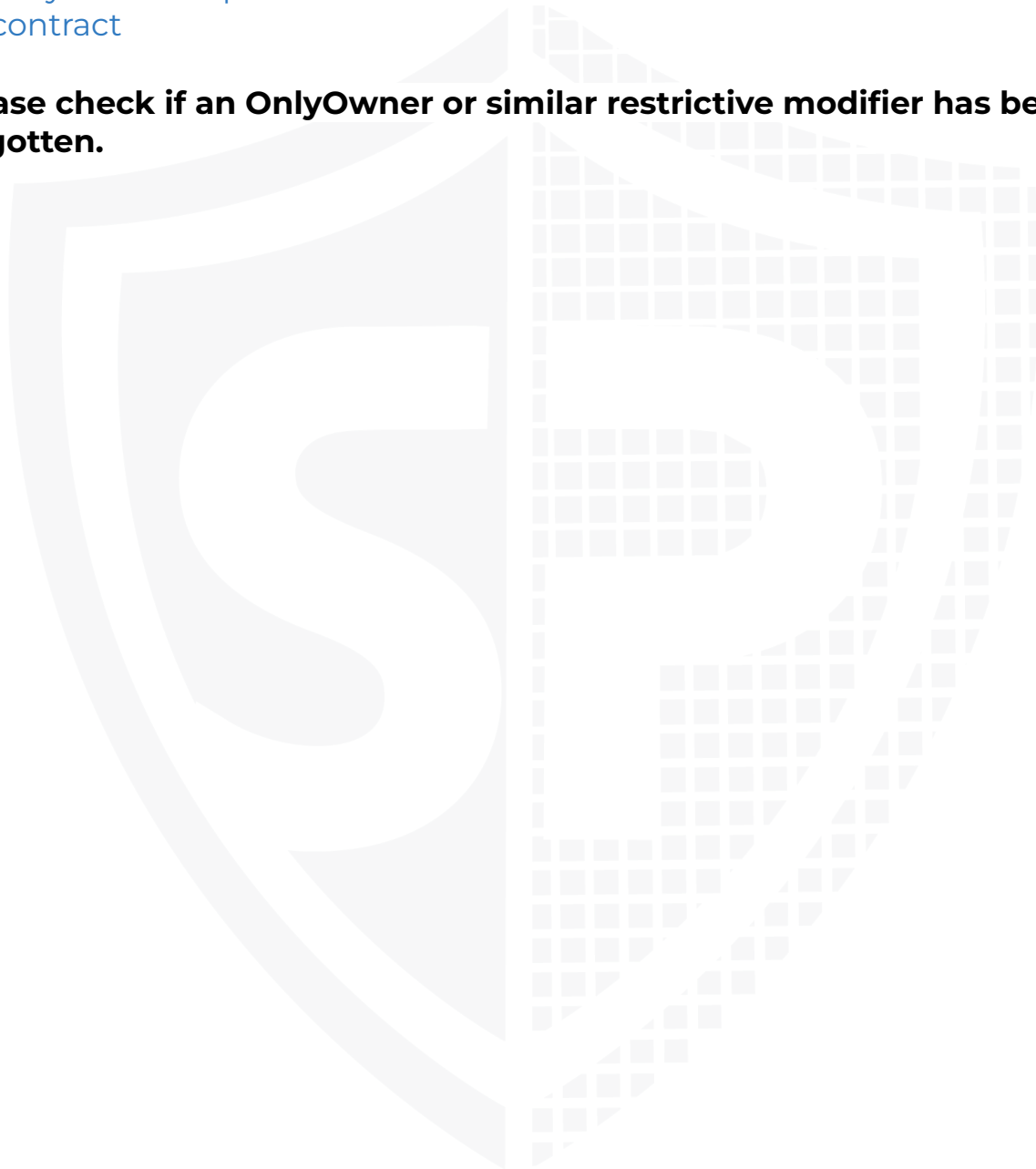
Note: Functions from imported libraries were not listed here.

## Comments

- Existing Modifiers
  - onlyAllowedOperator
  - onlyAllowedOperatorApproval
  - onlyOwner
  - whenNotPaused

- initializer
- Owner can
  - Create new presaleRounds
  - Pause contract
  - Change Upcoming Rounds
  - Set Payout Registry address
- Only allowed operators are allowed to transfer ERC721 tokens from the contract





**Please check if an OnlyOwner or similar restrictive modifier has been forgotten.**





# Source Units in Scope

## v1.2

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/VitreusVoucher.sol	1	————	1093	1042	458	445	461	
	Totals	1	————	1093	1042	458	445	461	

### Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalised lines of the source unit (e.g. normalises functions spanning multiple lines)
nSLOC	normalised source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

# Audit Results

## Critical issues

**No critical issues**

## High issues

**High issues found**

Issue	File	Type	Line	Description	Status
#1	Main	USDC Receiver	115	If the msg.sender is the _usdcReceiver address the msg.sender is able to mint tokens for free. We recommend you to check whether the caller is the _usdcReceiver.  Same for the deposit	Fixed
#2	Main	Initialize all the variables	191	_usdc is never initialized but the contract is calling the address. Initialize the variable in the "initialize" function as well.	Fixed
#3	Main	The owner is able to mint new tokens	424	The owner is able to mint new tokens. Also when the contract is paused. That means that the owner is excluded to mint new tokens.	Acknowledged

## Medium issues

**No medium issues**

## Low issues

Issue	File	Type	Line	Description	Status
#1	Main	A floating pragma is set	2	The current pragma Solidity directive is „^0.8.9“.	Fixed
#2	Main	Missing Zero Address Validation (missing-zero-check)	67	Check that the address is not zero	Fixed

#3	Main	State variable visibility is not set	30, 31, 33, 34, 35, 36, 51, 52, 57	It is best practice to set the visibility of state variables explicitly	Fixed
#4	Main	Local variables shadowing	354, 299, 360, 137, 201, 159, 207	Rename the local variables that shadow another component. We recommend you to modify "owner" to something similar like "owner_".	Fixed
#5	Main	Allocation can be set to 0	296	Since the allocation can be set to 0, the availableDeposit amount will be also 0 in L545, L640	Fixed

## Informational issues

Issue	File	Type	Line	Description	Status
#1	Main	Functions that are not used	251, 304, 309	Remove unused functions.  Before removing check the function, it could be possible, that you forget to implement it into the contract	Fixed
#2	Main	Unused state variables	30	Remove unused state variables	Fixed
#3	Main	Pause function	See description	The pause functionality is not implemented in the contract but the "whenPaused" modifier was used. Implement pause functionality for the owner only or remove the modifier.	Fixed
#4	Main	Misspelling	183	It is recommended to adjust the misspelling in the contract.	Fixed
#5	Main	Check for Zero value	195-199, 205, 212	Checking the msg.value is zero is recommended. Otherwise, the contract should not call the deposit/mint function in general.	Fixed

#6	Main	Disable initialize	140	<p>The disableInitializers function in the constructor should only be called after the first version of the upgradeable was deployed already. Otherwise the owner cannot call the “initialize” function.</p> <p>Make sure to implement this function back to the contract before updating the contract implementation.</p>	Fixed
----	------	--------------------	-----	--	-------

## Audit Comments

### 30. August 2023:

- The owner can deploy a new version of the contract which can change any limit and give the owner new privileges
- The Payout Registry interface which is used in the contract was not provided to us in the contract scope so we cannot comment on its security.
- Read the whole report and modifiers section for more information

## SWC Attacks

ID	Title	Relationships	Status
<a href="#">SW C-1 36</a>	Unencrypted Private Data On-Chain	<a href="#">CWE-767: Access to Critical Private Variable via Public Method</a>	PASSED
<a href="#">SW C-1 35</a>	Code With No Effects	<a href="#">CWE-1164: Irrelevant Code</a>	PASSED
<a href="#">SW C-1 34</a>	Message call with hardcoded gas amount	<a href="#">CWE-655: Improper Initialization</a>	PASSED
<a href="#">SW C-1 33</a>	Hash Collisions With Multiple Variable Length Arguments	<a href="#">CWE-294: Authentication Bypass by Capture-replay</a>	PASSED
<a href="#">SW C-1 32</a>	Unexpected Ether balance	<a href="#">CWE-667: Improper Locking</a>	PASSED
<a href="#">SW C-1 31</a>	Presence of unused variables	<a href="#">CWE-1164: Irrelevant Code</a>	PASSED
<a href="#">SW C-1 30</a>	Right-To-Left-Override control character (U+202E)	<a href="#">CWE-451: User Interface (UI) Misrepresentation of Critical Information</a>	PASSED
<a href="#">SW C-1 29</a>	Typographical Error	<a href="#">CWE-480: Use of Incorrect Operator</a>	PASSED
<a href="#">SW C-1 28</a>	DoS With Block Gas Limit	<a href="#">CWE-400: Uncontrolled Resource Consumption</a>	PASSED

<a href="#">SW C-1 27</a>	Arbitrary Jump with Function Type Variable	<a href="#">CWE-695: Use of Low-Level Functionality</a>	<b>PASSED</b>
<a href="#">SW C-1 25</a>	Incorrect Inheritance Order	<a href="#">CWE-696: Incorrect Behavior Order</a>	<b>PASSED</b>
<a href="#">SW C-1 24</a>	Write to Arbitrary Storage Location	<a href="#">CWE-123: Write-what-where Condition</a>	<b>PASSED</b>
<a href="#">SW C-1 23</a>	Requirement Violation	<a href="#">CWE-573: Improper Following of Specification by Caller</a>	<b>PASSED</b>
<a href="#">SW C-1 22</a>	Lack of Proper Signature Verification	<a href="#">CWE-345: Insufficient Verification of Data Authenticity</a>	<b>PASSED</b>
<a href="#">SW C-1 21</a>	Missing Protection against Signature Replay Attacks	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	<b>PASSED</b>
<a href="#">SW C-1 20</a>	Weak Sources of Randomness from Chain Attributes	<a href="#">CWE-330: Use of Insufficiently Random Values</a>	<b>PASSED</b>
<a href="#">SW C-11 9</a>	Shadowing State Variables	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>
<a href="#">SW C-11 8</a>	Incorrect Constructor Name	<a href="#">CWE-665: Improper Initialization</a>	<b>PASSED</b>
<a href="#">SW C-11 7</a>	Signature Malleability	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	<b>PASSED</b>

<a href="#">SW C-11 6</a>	Timestamp Dependence	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 5</a>	Authorization through tx.origin	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 4</a>	Transaction Order Dependence	<a href="#">CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')</a>	<b>PASSED</b>
<a href="#">SW C-11 3</a>	DoS with Failed Call	<a href="#">CWE-703: Improper Check or Handling of Exceptional Conditions</a>	<b>PASSED</b>
<a href="#">SW C-11 2</a>	Delegatecall to Untrusted Callee	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 1</a>	Use of Deprecated Solidity Functions	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 0</a>	Assert Violation	<a href="#">CWE-670: Always-Incorrect Control Flow Implementation</a>	<b>PASSED</b>
<a href="#">SW C-1 09</a>	Uninitialized Storage Pointer	<a href="#">CWE-824: Access of Uninitialized Pointer</a>	<b>PASSED</b>
<a href="#">SW C-1 08</a>	State Variable Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>
<a href="#">SW C-1 07</a>	Reentrancy	<a href="#">CWE-841: Improper Enforcement of Behavioral Workflow</a>	<b>PASSED</b>
<a href="#">SW C-1 06</a>	Unprotected SELFDESTRUCT Instruction	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>

<a href="#">SW C-1 05</a>	Unprotected Ether Withdrawal	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>
<a href="#">SW C-1 04</a>	Unchecked Call Return Value	<a href="#">CWE-252: Unchecked Return Value</a>	<b>PASSED</b>
<a href="#">SW C-1 03</a>	Floating Pragma	<a href="#">CWE-664: Improper Control of a Resource Through its Lifetime</a>	<b>PASSED</b>
<a href="#">SW C-1 02</a>	Outdated Compiler Version	<a href="#">CWE-937: Using Components with Known Vulnerabilities</a>	<b>PASSED</b>
<a href="#">SW C-1 01</a>	Integer Overflow and Underflow	<a href="#">CWE-682: Incorrect Calculation</a>	<b>PASSED</b>
<a href="#">SW C-1 00</a>	Function Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>



*Solid  
Proofed*

**Blockchain Security | Smart Contract Audits | KYC  
Development | Marketing**

  
MADE IN GERMANY