SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

Revolt 2 Earn

Audit

Security Assessment
15. June, 2022

For

 SolidProof_io   **@solidproof_io**

Disclaimer

Version	Date	Description
1.0	10. June 2022	• Layout project • Automated- /Manual-Security Testing • Summary
1.1	14. June 2022	• Reaudit
1.2	15. June 2022	• Reaudit

Network
Polygon Matic

Website
https://revolt.cultdao.io/

Telegram
https://t.me/revolt2earn

Twitter
https://twitter.com/wearecultdao

Medium
https://wearecultdao.medium.com/

Discord
https://discord.com/invite/hHDBvNnXqe

Reddit
http://reddit.com/r/cultdao/

Description

Revolt (RVLT) is CULT DAOs first ecosystem token, and has been built on the Polygon network as a Polygon POS token with 0.4% taxation. Whereas CULT works to fund protocols furthering decentralization, RVLT seeks to support The Many individuals who are working towards the same goal.

Each 15 days 490 stakers (+10 consistent NFT owners) are picked randomly (through Chainlink VRF) from all RVLT stakers (uRVLT owners).

These 500 (known as the CULTmanders) have the job of approving or disapproving submissions by the users of the actions they have taken to help the CULT ecosystem, whether this be stickering, leafleting, shilling or anything else, they are effectively being paid to further the cause of decentralization and the ecosystem.

Project Engagement

During the 9th of June 2022, **Revolt2Earn 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

Contract Link
v1.2
- Github
    - https://github.com/cultdao-developer/revolt2earn
    - Commit: 8986c15078592e74ee876c5898c798286ec50976

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)
Critical	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.
High	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 aspossible.
Medium	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.
Low	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.
Informational	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
@chainlink/contracts/src/v0.8/interfaces/VRFCoordinatorV2Interface.sol	1
@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol	4
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	6
@openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol	6
@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol	2
@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20VotesCompUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20VotesUpgradeable.sol	2
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/draft-ERC20PermitUpgradeable.sol	2
@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC721/IERC721Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/CountersUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/math/SafeMathUpgradeable.sol	5

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.

v1.0

File Name	SHA-1 Hash
contracts/treasury.sol	cc9de1fe66a54dc55f83f9d6434cf8da40bdc14f
contracts/GovernorBravoInterfaces.sol	2cf173cc8a8a80183497da32081b0af990733a24
contracts/rvlt.sol	c18c733db504c0e9dc12cded483dcfcbd21f5125
contracts/timelock.sol	29ecb767a459651586a3308aebfddde1abcdb073
contracts/governance.sol	29cc8ed73ce52c1ec973f5e29ae125302695d1df
contracts/uRevolt.sol	9eb9a533e1eb268fee43ec98010561aa53cc0614
contracts/RandomNumberGenerator.sol	fe938f5ef6a2a5472d404721426e4e00ec2d8abe

Metrics

Source Lines
v1.0
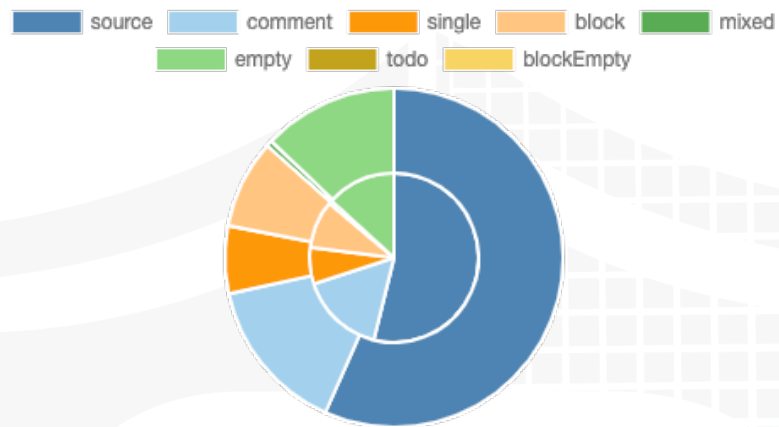
Risk Level
v1.0
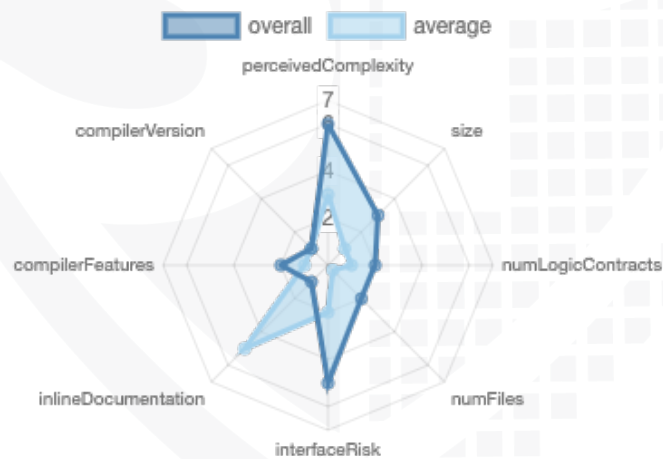
Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	11	0	9	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	93	5

Version	External	Internal	Private	Pure	View
1.0	68	113	0	8	27

State Variables

Version	Total	Public
1.0	89	83

Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`0.8.2` `^0.8.0`	`ABIEncoderV2`	`yes`	`yes` (1 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	EC Recover	New/Create/Create2
1.0	yes			yes	yes	

Inheritance Graph
v1.0

CallGraph
v1.0
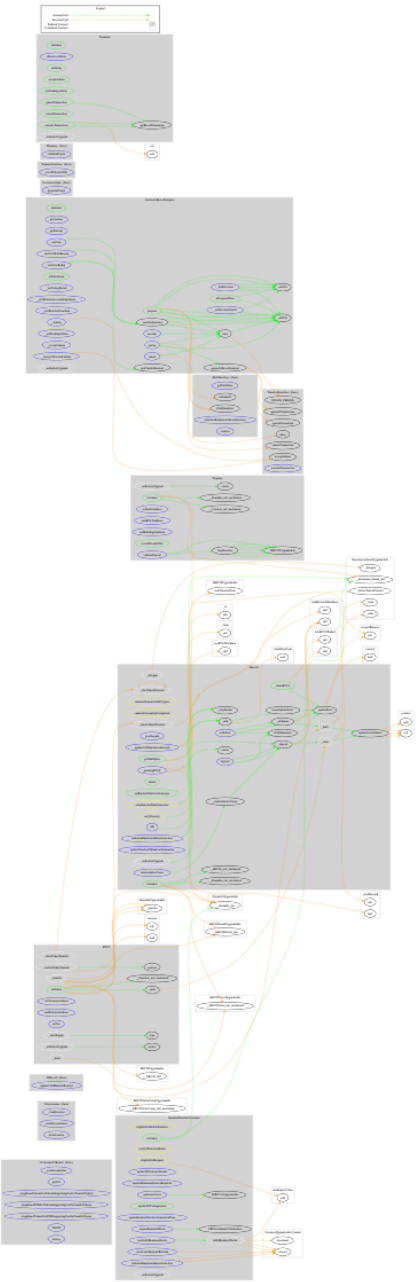
Scope of Work/Verify Claims

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).
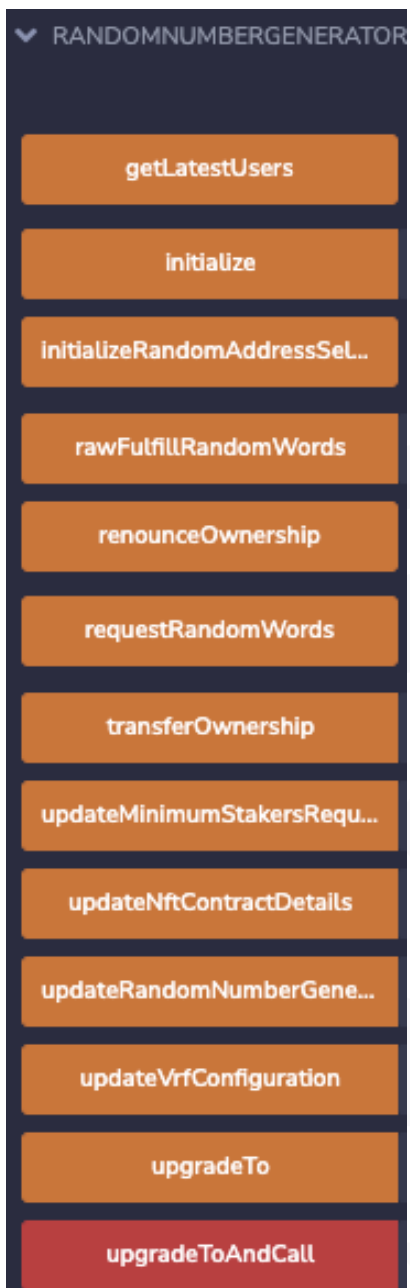
We will verify the following claims:
1. Overall checkup (Smart Contract Security)

Write functions of contract v1.0

RVLT
- approve
- decreaseAllow...
- delegate
- delegateBySig
- increaseAllow...
- initialize
- permit
- renounceOwn...
- setTax
- setTreasuryAd...
- setWhitelistA...
- transfer
- transferFrom
- transferOwner...
- upgradeTo
- upgradeToAnd...

TIMELOCK)
- acceptAdmin
- cancelTransact...
- executeTransa...
- initialize
- queueTransact...
- setDelay
- setPendingAd...
- upgradeTo
- upgradeToAnd...

UREVOLT
- add
- admin
- approve
- claimRVLT
- decreaseAllow...
- delegate
- delegateBySig
- deposit
- increaseAllow...
- initialize
- initializeRando...
- massUpdateP...
- permit
- randomSelect...
- renounceOwn...
- setRandomNu...
- transfer
- transferFrom
- transferOwner...
- updateCultMa...
- updateNumbe...
- updatePool
- upgradeTo
- upgradeToAnd...
- withdraw

TREASURY
- initialize
- renounceOwn...
- setDAOAddress
- setMultiSignA...
- setuRVLTAddr...
- transferOwner...
- upgradeTo
- upgradeToAnd...
- validatePayout

GOVERNORBRAVODELEGATE
- _acceptAdmin
- _AcceptTimelockAdmin
- _fundInvestee
- _initiate
- _setInvesteeDetails
- _setPendingAdmin
- _setVotingPeriod
- _setWhitelistAccountExpirat...
- _setWhitelistGuardian
- cancel
- castVote
- castVoteBySig
- castVoteWithReason
- execute
- initialize
- propose
- queue
- upgradeTo
- upgradeToAndCall

RANDOMNUMBERGENERATOR

- getLatestUsers
- initialize
- initializeRandomAddressSeL...
- rawFulfillRandomWords
- renounceOwnership
- requestRandomWords
- transferOwnership
- updateMinimumStakersRequ...
- updateNftContractDetails
- updateRandomNumberGene...
- updateVrfConfiguration
- upgradeTo
- upgradeToAndCall

Overall checkup (Smart Contract Security)

Tested	Verified
✓	✓

Legend

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

Modifiers and public functions
v1.0

uRevolt

- ∨ ⬥ initialize
  - ⊚ initializer
- ∨ ⬥ add
  - ⊚ onlyOwner
- ∨ ⬥ massUpdatePools
  - ⊚ randomGenerationCompleted
  - ⬥ updatePool
- ∨ ⬥ deposit
  - ⊚ randomGenerationCompleted
  - ⊚ nonReentrant
- ∨ ⬥ withdraw
  - ⊚ randomGenerationCompleted
  - ⊚ nonReentrant
- ∨ ⬥ claimRVLT
  - ⊚ randomGenerationCompleted
  - ⊚ nonReentrant
  - ⬥ admin
- ∨ ⬥ setRandomNumberGenerator
  - ⊚ onlyOwner
- ∨ ⬥ updateCultMandatorsReward
  - ⊚ onlyTreasury
- ∨ ⬥ updateNumberOfRandomGeneration
  - ⊚ onlyOwner
- ∨ ⬥ initializeRandomAddressSelection
  - ⊚ onlyRandomNumGenerator
  - ⊚ nonReentrant
- ∨ ⬥ randomSelectUsers
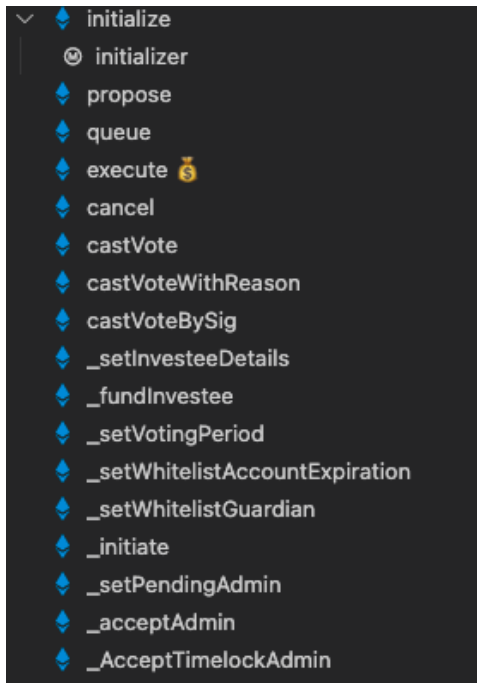 - ⊚ randomGenerationInProgress
  - ⊚ nonReentrant

Treasury

- ∨ ⬥ initialize
  - ⊚ initializer
- ∨ ⬥ setDAOAddress
  - ⊚ onlyOwner
- ∨ ⬥ setuRVLTAddress
  - ⊚ onlyOwner
- ∨ ⬥ setMultiSignAddress
  - ⊚ onlyOwner
- ⬥ validatePayout
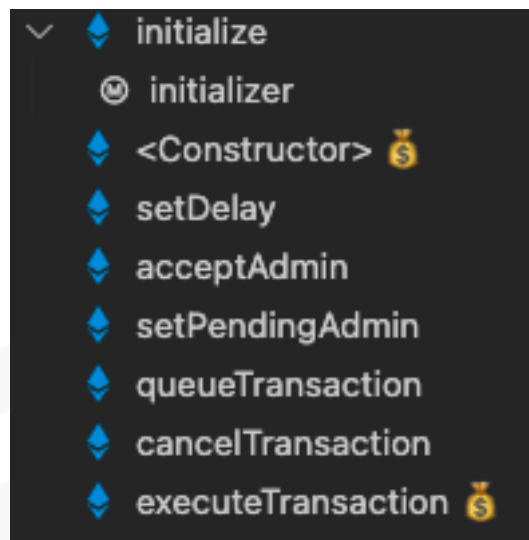
Timelock

- ∨ ⬥ initialize
  - ⊚ initializer
- ⬥ <Constructor> 💰
- ⬥ setDelay
- ⬥ acceptAdmin
- ⬥ setPendingAdmin
- ⬥ queueTransaction
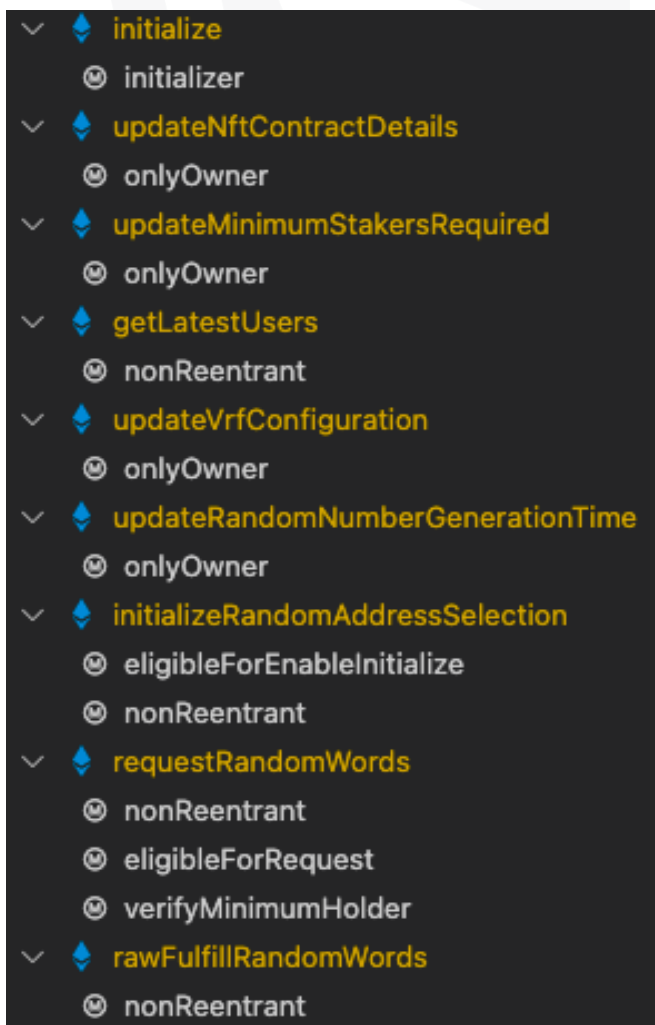- ⬥ cancelTransaction
- ⬥ executeTransaction 💰

Governance

- ∨ ♦ initialize
  - ◎ initializer
- ♦ propose
- ♦ queue
- ♦ execute 💰
- ♦ cancel
- ♦ castVote
- ♦ castVoteWithReason
- ♦ castVoteBySig
- ♦ _setInvesteeDetails
- ♦ _fundInvestee
- ♦ _setVotingPeriod
- ♦ _setWhitelistAccountExpiration
- ♦ _setWhitelistGuardian
- ♦ _initiate
- ♦ _setPendingAdmin
- ♦ _acceptAdmin
- ♦ _AcceptTimelockAdmin

Revolt

- ∨ ♦ initialize
  - ◎ initializer
- ♦ <Constructor> 💰
- ♦ setDelay
- ♦ acceptAdmin
- ♦ setPendingAdmin
- ♦ queueTransaction
- ♦ cancelTransaction
- ♦ executeTransaction 💰

RandomNumberGenerator

- ∨ ♦ initialize
  - ◎ initializer
- ∨ ♦ updateNftContractDetails
  - ◎ onlyOwner
- ∨ ♦ updateMinimumStakersRequired
  - ◎ onlyOwner
- ∨ ♦ getLatestUsers
  - ◎ nonReentrant
- ∨ ♦ updateVrfConfiguration
  - ◎ onlyOwner
- ∨ ♦ updateRandomNumberGenerationTime
  - ◎ onlyOwner
- ∨ ♦ initializeRandomAddressSelection
  - ◎ eligibleForEnableInitialize
  - ◎ nonReentrant
- ∨ ♦ requestRandomWords
  - ◎ nonReentrant
  - ◎ eligibleForRequest
  - ◎ verifyMinimumHolder
- ∨ ♦ rawFulfillRandomWords
  - ◎ nonReentrant

Note: Not listed functions/modifiers was imported from external libraries

Comments

- *Deployer can set following state variables without any limitations*
    - Governance
        - votingPeriod
        - investee._investee
   - investee._fundAmount
        - whitelistAccountExpirations
    - uRevolt
        - randomThreshold
    - Revolt
        - tax
    - RandomNumberGenerator
        - tokenIds
        - minimumStakersRequired
        - callbackGasLimit
            - Max 2^32 - 1
        - requestConfirmations
            - Max 2^16 - 1
        - _numWords
            - Max 2^32 - 1
        - _subscriptionId
            - Max 2^64 - 1

- *Deployer can enable/disable following state variables*
    - Revolt
        - whitelistedAddress

- *Deployer can set following addresses*
    - Governance
        - whitelistGuardian
        - pendingAdmin
    - uRevolt
        - adminAddress
        - numberGenerator
    - Treasury
        - dao
        - uRvlt
        - multSignWallet
    - Timelock
        - admin
        - pendingAdmin
    - Revolt
        - treasury

- RandomNumberGenerator
    - nftAddress
    - vrfCoordinator
    - keyHash
    -
- *Existing Modifiers*
    - eligibleForEnableInitialize
    - eligibleForRequest
    - verifyMinimumHolder
    - onlyTreasury
    - onlyRandomNumGenerator
    - randomGenerationInProgress
    - randomGenerationCompleted

- uRevolt
    - Owner can add new poolinfo
    - Treasury can update totalRewardMandators/ totalRVLTRewardMandators without limitations

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

Source Units in Scope
v1.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝🔍	contracts/treasury.sol	2	3	152	96	84	1	132	💰🔻
📝🔍	contracts/GovernorBravoInterfaces.sol	4	4	207	189	75	62	52	✏️💰
📝🔍	contracts/rvlt.sol	1	1	113	90	74	1	67	————
📝	contracts/timelock.sol	1	————	126	126	92	3	87	💰🎰
📝	contracts/governance.sol	1	————	453	453	253	134	227	💻✏️💰🎰✏️
📝	contracts/uRevolt.sol	1	————	469	446	358	45	281	🎰
📝🔍	contracts/RandomNumberGenerator.sol	1	1	292	247	137	76	101	————
📝🔍	**Totals**	**11**	**9**	**1812**	**1647**	**1073**	**322**	**947**	💻✏️💰🔻🎰✏️

Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
nSLOC	normalized 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

AUDIT PASSED

Critical issues

No critical issues

High issues

No high issues

Medium issues

No medium issues

Low issues

No low issues

Informational issues

No informational issues

Audit Comments

We recommend you to use the special form of comments (NatSpec Format, Follow link for more information https://docs.soliditylang.org/en/v0.5.10/natspec-format.html) for your contracts to provide rich documentation for functions, return variables and more. This helps investors to make clear what that variables, functions etc. do.

11. June 2022:

- Owner can deploy a new version of the contract which can change any limit and give owner new privileges
- Read whole report and modifiers section for more information

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	**PASSED**
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	**PASSED**
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	**PASSED**
SWC-133	Hash Collisions With Multiple Variable Length Arguments	CWE-294: Authentication Bypass by Capture-replay	**PASSED**
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	**PASSED**
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	**PASSED**
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	**PASSED**
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	**PASSED**
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	**PASSED**

[SWC-127](#)	Arbitrary Jump with Function Type Variable	[CWE-695: Use of Low-Level Functionality](#)	**PASSED**
[SWC-125](#)	Incorrect Inheritance Order	[CWE-696: Incorrect Behavior Order](#)	**PASSED**
[SWC-124](#)	Write to Arbitrary Storage Location	[CWE-123: Write-what-where Condition](#)	**PASSED**
[SWC-123](#)	Requirement Violation	[CWE-573: Improper Following of Specification by Caller](#)	**PASSED**
[SWC-122](#)	Lack of Proper Signature Verification	[CWE-345: Insufficient Verification of Data Authenticity](#)	**PASSED**
[SWC-121](#)	Missing Protection against Signature Replay Attacks	[CWE-347: Improper Verification of Cryptographic Signature](#)	**PASSED**
[SWC-120](#)	Weak Sources of Randomness from Chain Attributes	[CWE-330: Use of Insufficiently Random Values](#)	**PASSED**
[SWC-119](#)	Shadowing State Variables	[CWE-710: Improper Adherence to Coding Standards](#)	**PASSED**
[SWC-118](#)	Incorrect Constructor Name	[CWE-665: Improper Initialization](#)	**PASSED**
[SWC-117](#)	Signature Malleability	[CWE-347: Improper Verification of Cryptographic Signature](#)	**PASSED**

[SWC-116](#)	Timestamp Dependence	[CWE-829: Inclusion of Functionality from Untrusted Control Sphere](#)	**PASSED**
[SWC-115](#)	Authorization through tx.origin	[CWE-477: Use of Obsolete Function](#)	**PASSED**
[SWC-114](#)	Transaction Order Dependence	[CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')](#)	**PASSED**
[SWC-113](#)	DoS with Failed Call	[CWE-703: Improper Check or Handling of Exceptional Conditions](#)	**PASSED**
[SWC-112](#)	Delegatecall to Untrusted Callee	[CWE-829: Inclusion of Functionality from Untrusted Control Sphere](#)	**PASSED**
[SWC-111](#)	Use of Deprecated Solidity Functions	[CWE-477: Use of Obsolete Function](#)	**PASSED**
[SWC-110](#)	Assert Violation	[CWE-670: Always-Incorrect Control Flow Implementation](#)	**PASSED**
[SWC-109](#)	Uninitialized Storage Pointer	[CWE-824: Access of Uninitialized Pointer](#)	**PASSED**
[SWC-108](#)	State Variable Default Visibility	[CWE-710: Improper Adherence to Coding Standards](#)	**PASSED**
[SWC-107](#)	Reentrancy	[CWE-841: Improper Enforcement of Behavioral Workflow](#)	**PASSED**
[SWC-106](#)	Unprotected SELFDESTRUCT Instruction	[CWE-284: Improper Access Control](#)	**PASSED**

[SWC-105](#)	Unprotected Ether Withdrawal	[CWE-284: Improper Access Control](#)	**PASSED**
[SWC-104](#)	Unchecked Call Return Value	[CWE-252: Unchecked Return Value](#)	**PASSED**
[SWC-103](#)	Floating Pragma	[CWE-664: Improper Control of a Resource Through its Lifetime](#)	**PASSED**
[SWC-102](#)	Outdated Compiler Version	[CWE-937: Using Components with Known Vulnerabilities](#)	**PASSED**
[SWC-101](#)	Integer Overflow and Underflow	[CWE-682: Incorrect Calculation](#)	**PASSED**
[SWC-100](#)	Function Default Visibility	[CWE-710: Improper Adherence to Coding Standards](#)	**PASSED**