Smart Contract Security Audit V1

Dung Vesting Smart Contract Audit

https://dungtoken.com/

April 19, 2023

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

Table of Contents

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Website**: https://dungtoken.com/
- **Twitter**: https://twitter.com/TokenDung
- **Telegram**: https://t.me/Dung_Token
- **Whitepaper**: https://dungtoken.com/assets/dungtoken_whitepaper.c3827862.pdf
- **discord**: https://discord.io/dung
- **reddit**: https://www.reddit.com/u/Dung_Token
- **Platform**: Polygon
- **Contract Address**: 0xcc910bed75a7b43240cfdfe91d93cb1d75cbb89f
- **Code Source**: https://polygonscan.com/address/0xcc910bed75a7b43240cfdfe91d93cb1d75cbb89f#code
- **Audit Report**: https://github.com/Saferico/DUNG-Vesting-smart-contract-Audit

Smart Contract Information:

- **Name:** DUNG Vesting smart contract
- **Team comment:** Our tokens are locked in a smart contract.

We decided vesting is the best choice to promote growth and Sustainability.

- **Total supply of the token:** 21.000.000.000.000

Contracts address deployed to test net (Polygon)
Dung Vesting smart contracts on Polygon test-net by the auditor to test every function .

https://mumbai.polygonscan.com/address/0x747994f6f9f3be267ce34c2e0b1f3eec1bcc59f5

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Secured**.

Secured	✓
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 1 low, 0 very low-level issues and 2 notes in all solidity files of the contract

The files:

Vesting.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
vesting.sol	**a563a946cd0371e90d9a5489fcfd4449620df526**	0xcc910bed75a7b43240cfdfe91d93cb1d75cbb89f

- Contract: vesting
- Inherit: Ownable, ReentrancyGuard
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
getClaimableAmount	✓	Read / public	**Passed**
locked	✓	Read / public	**Passed**
numberOfSchedules	✓	Read / public	**Passed**
schedules	✓	Read / public	**Passed**
owner	✓	Read / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
createMultiSchedule	✓	Write / public	**Passed**
createSchedule	✓	Write / public	**Passed**
cancel	✓	Write / public	**Passed**
claim	✓	Write / public	**Passed**
withdraw	✓	Write / public	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**Passed**
2	Race conditions and Reentrancy. Cross-function race conditions.	**Passed**
3	Possible delays in data delivery.	**Passed**
4	Oracle calls.	**Passed**
5	Design Logic.	**Passed**
6	Timestamp dependence.	**Passed with notes**
7	Integer Overflow and Underflow.	**Passed**
8	DoS with Revert.	**Passed**
9	DoS with block gas limit.	**Passed with notes**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**Passed**
12	The impact of the exchange rate on the logic.	**Passed**
13	Private user data leaks.	**Passed**
14	Malicious Event log.	**Passed**
15	Scoping and Declarations.	**Passed**
16	Uninitialized storage pointers.	**Passed**
17	Arithmetic accuracy.	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can’t lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can’t have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can’t affect smart contract execution and can be ignored.

Audit Findings

Critical:

No Critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found.

Medium:

No Medium severity vulnerabilities were found.

Low:

#Use of block.timestamp for comparisons

Description

"block.timestamp" can be influenced by miners to a certain degree. That mean that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

Status: Acknowledged, It is usual for Vesting contracts.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Unnecessary import of IERC20 library

Description

The main contract inherits: Ownable, and ReentrancyGuard. SafeERC20 which is already

import IERC20 library, so no need to import it again in the main contract.

Remediation

Remove unnecessary library from the main contract save some gas fees.

Status: Acknowledged

#Compiler version is old

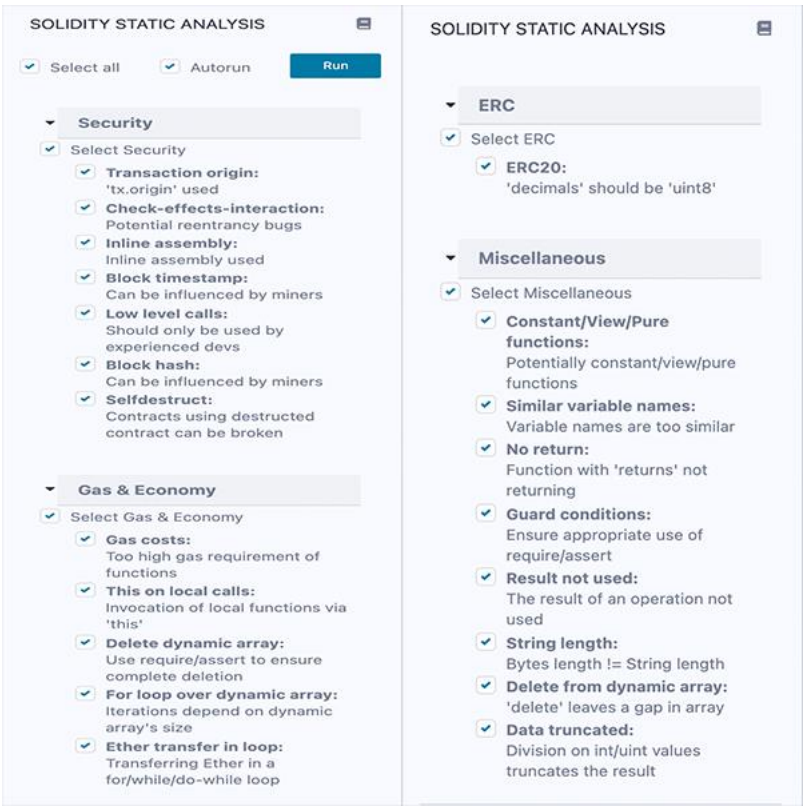
Description

The compiler being used was released 3 years ago. It's recommended to use more recent compiler version, there can be benefits like reduction in bytecode size etc.

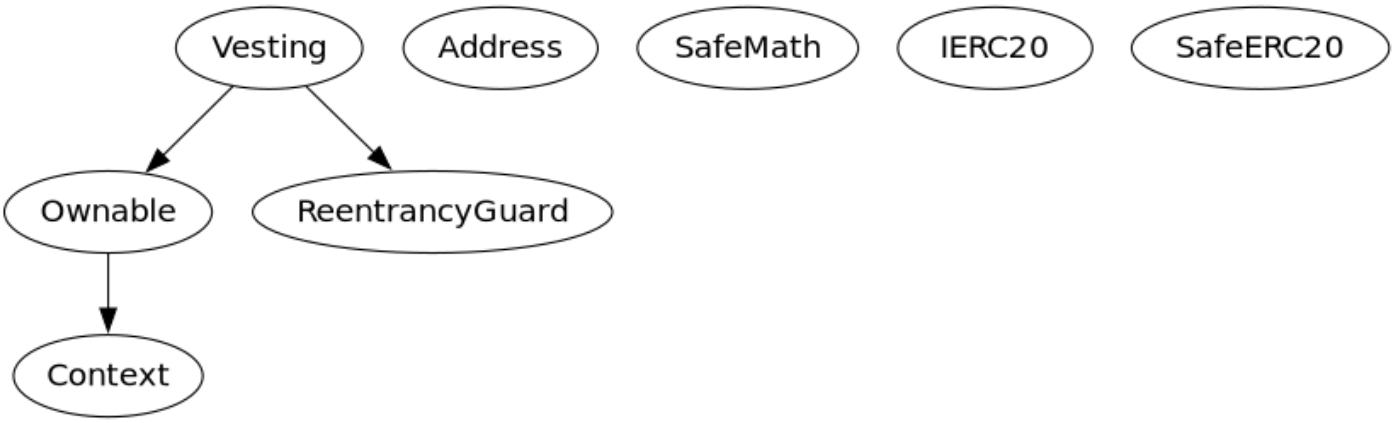
Status: Acknowledged.

Automatic Testing

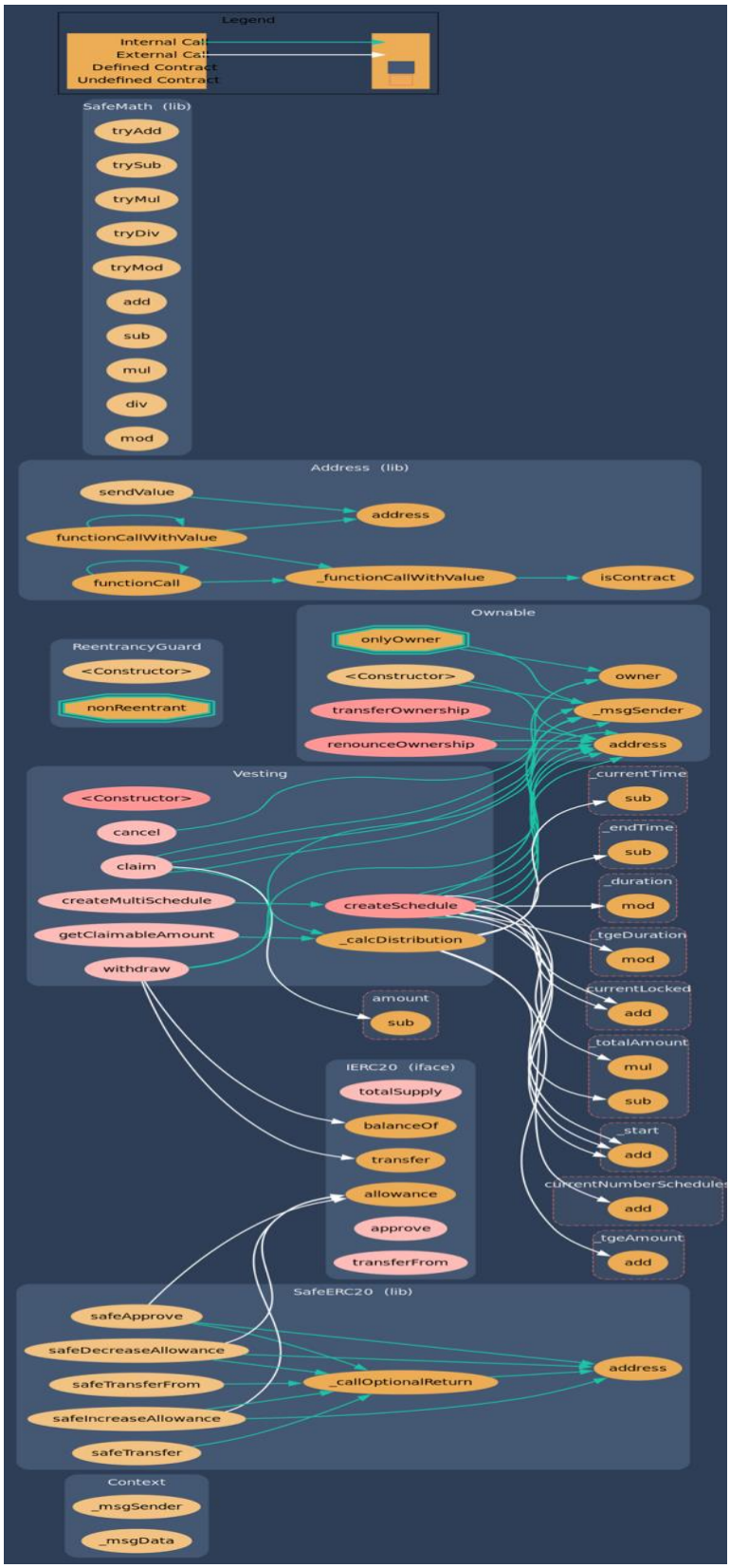
1- SOLIDITY STATIC ANALYSIS



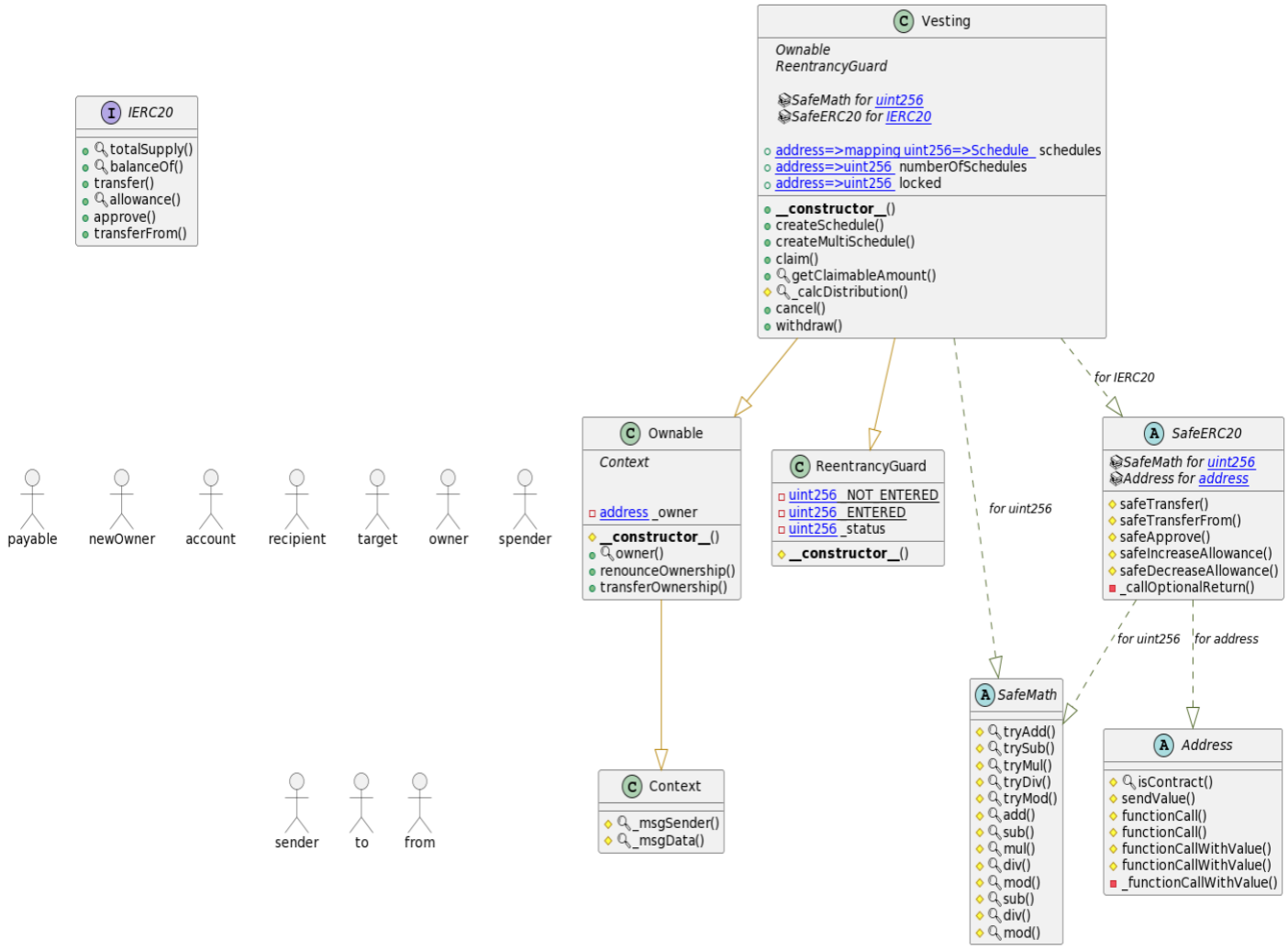
2- Inheritance graph



3- Call graph


Legend
Internal Cal
External Ca
Defined Contract
Undefined Contract
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
add
sub
mul
div
mod
Address (lib)
sendValue
address
functionCallWithValue
functionCall
_functionCallWithValue
isContract
Ownable
onlyOwner
<Constructor>
transferOwnership
renounceOwnership
owner
_msgSender
address
ReentrancyGuard
<Constructor>
nonReentrant
Vesting
<Constructor>
cancel
claim
createMultiSchedule
getClaimableAmount
withdraw
createSchedule
_calcDistribution
_currentTime
sub
_endTime
sub
_duration
mod
tgeDuration
mod
currentLocked
add
totalAmount
mul
sub
_start
add
currentNumberSchedules
add
_tgeAmount
add
amount
sub
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
SafeERC20 (lib)
safeApprove
safeDecreaseAllowance
safeTransferFrom
safeIncreaseAllowance
safeTransfer
_callOptionalReturn
address
Context
_msgSender
_msgData

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
24a084df  =>  sendValue(address,uint256)
a0b5ffb0  =>  functionCall(address,bytes)
241b5886  =>  functionCall(address,bytes,string)
2a011594  =>  functionCallWithValue(address,bytes,uint256)
d525ab8a  =>  functionCallWithValue(address,bytes,uint256,string)
36455e42  =>  _functionCallWithValue(address,bytes,uint256,string)
884557bf  =>  tryAdd(uint256,uint256)
a29962b1  =>  trySub(uint256,uint256)
6281efa4  =>  tryMul(uint256,uint256)
736ecb18  =>  tryDiv(uint256,uint256)
38dc0867  =>  tryMod(uint256,uint256)
771602f7  =>  add(uint256,uint256)
b67d77c5  =>  sub(uint256,uint256)
c8a4ac9c  =>  mul(uint256,uint256)
a391c15b  =>  div(uint256,uint256)
f43f523a  =>  mod(uint256,uint256)
e31bdc0a  =>  sub(uint256,uint256,string)
b745d336  =>  div(uint256,uint256,string)
71af23e8  =>  mod(uint256,uint256,string)
18160ddd  =>  totalSupply()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
d0c407e1  =>  safeTransfer(IERC20,address,uint256)
5beae096  =>  safeTransferFrom(IERC20,address,address,uint256)
d6dcec8d  =>  safeApprove(IERC20,address,uint256)
390cc046  =>  safeIncreaseAllowance(IERC20,address,uint256)
5164ffed  =>  safeDecreaseAllowance(IERC20,address,uint256)
becc5a20  =>  _callOptionalReturn(IERC20,bytes)
99e6caed  =>
createSchedule(address,uint256,uint256,address,uint256,uint256,uint256,uint256,uint
256)
b0bf6ee3  =>
createMultiSchedule(address[],uint256[],uint256,address,uint256,uint256,uint256,uin
t256,uint256)
379607f5  =>  claim(uint256)
b3743f60  =>  getClaimableAmount(address,uint256)
ef9c6310  =>  _calcDistribution(Schedule)
98590ef9  =>  cancel(address,uint256)
00f714ce  =>  withdraw(uint256,address)
```

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/Vesting.sol	a563a946cd0371e90d9a5489fcfd4449620df526

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Internal 🔒	🛑	
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
ReentrancyGuard	Implementation			
└	<Constructor>	Internal 🔒	🛑	
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	_functionCallWithValue	Private 🔐	🛑	
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO❗

└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeERC20	Library		
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
└	_callOptionalReturn	Private 🔐	🛑	
Vesting	Implementation	Ownable, ReentrancyGuard	
└	<Constructor>	Public ❗	🛑	NO❗
└	createSchedule	Public ❗	🛑	onlyOwner
└	createMultiSchedule	External ❗	🛑	onlyOwner
└	claim	External ❗	🛑	nonReentrant
└	getClaimableAmount	External ❗		NO❗
└	_calcDistribution	Internal 🔒		
└	cancel	External ❗	🛑	nonReentrant onlyOwner
└	withdraw	External ❗	🛑	nonReentrant onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “Secured”.

✓ No mint function.
✓ No volatile code.
✓ No high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.