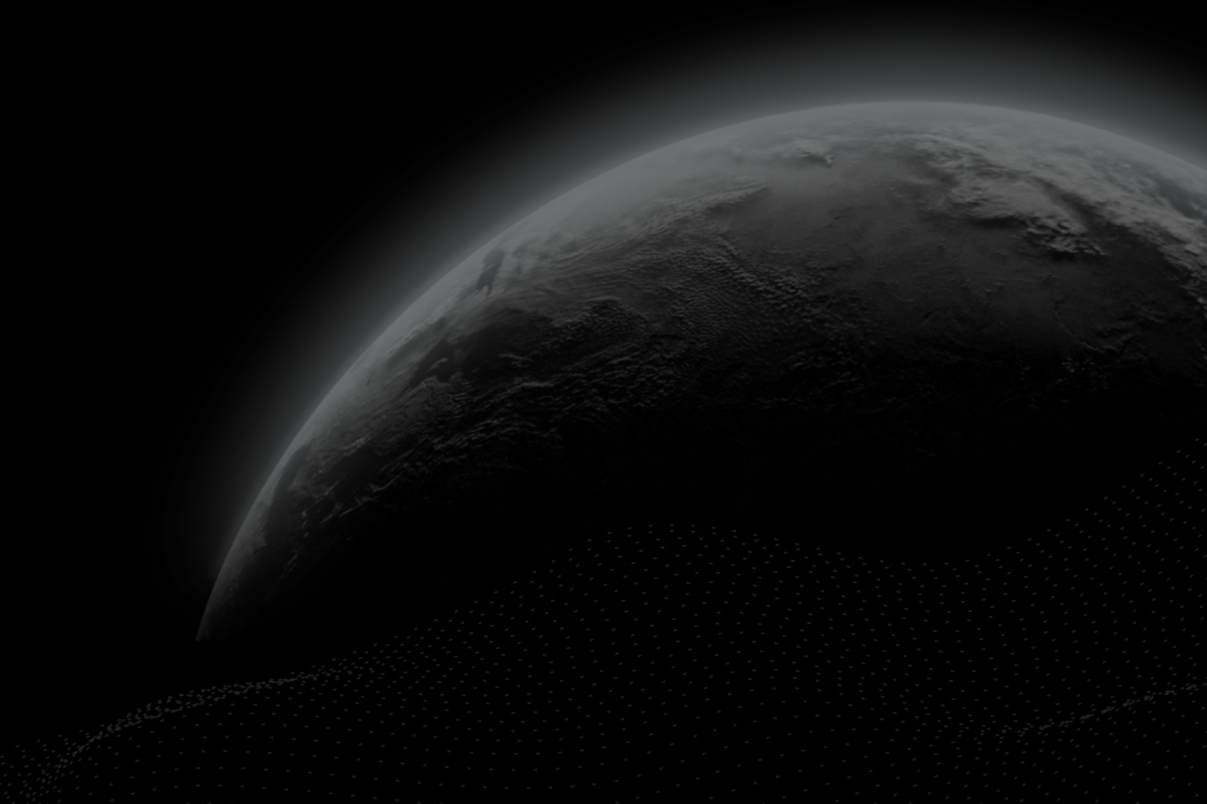
CERTIK

Security Assessment

Decentraland - PeriodicTokenVesting

CertiK Verified on Dec 2nd, 2022



CertiK Verified on Dec 2nd, 2022

Decentraland - PeriodicTokenVesting

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Ethereum	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 12/02/2022	N/A

CODEBASE

https://github.com/decentraland/vestings-builder

...View All

COMMITS

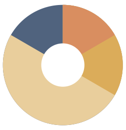
base: ad906381b96f23c0b82a9ba89cf3466914448139

update1: 243ee6e9abbc07a23d7abd0af30f464c03a7e59b

update2: a1e42ae8b71668040fb1700da28be49bf868cc03

...View All

Vulnerability Summary



6	2	0	0	4	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

🟥	0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧	1	Major	1 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨	1	Medium	1 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨	3	Minor	3 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦	1	Informational	1 Resolved	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | DECENTRALAND - PERIODICTOKENVESTING

CODEBASE | DECENTRALAND - PERIODICTOKENVESTING

Repository

https://github.com/decentraland/vestings-builder

Commit

base: ad906381b96f23c0b82a9ba89cf3466914448139

update1: 243ee6e9abbc07a23d7abd0af30f464c03a7e59b

update2: a1e42ae8b71668040fb1700da28be49bf868cc03

AUDIT SCOPE | DECENTRALAND - PERIODICTOKENVESTING

2 files audited ● 2 files with Acknowledged findings

ID	Repo	Commit	File	SHA256 Checksum
● PTV	decentraland/vestings-builder	ad90638	contracts/PeriodicTokenVesting.sol	c7244cbfd8201baf8e3b48650fb797e0e63fca877a480e803b8c9ea8ec43055c
● PER	decentraland/vestings-builder	a1e42ae	contracts/PeriodicTokenVesting.sol	77acd8de5e084393e67bc5f2acd48cc14d893bf53d8105caa2401c4cccad5fe6

APPROACH & METHODS | DECENTRALAND - PERIODICTOKENVESTING

This report has been prepared for Decentraland to discover issues and vulnerabilities in the source code of the Decentraland - PeriodicTokenVesting project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

REVIEW NOTES | DECENTRALAND - PERIODICTOKENVESTING

System Overview

The contract `PeriodicTokenVesting` is a customizable vesting schedule. At initialization, parameters such as the `beneficiary` address, the `token` to vest, the number of periods, the amount to vest in each period, the period length, the start time, and the cliff are specified. The vesting schedule also allows choice between a linear or lump sum releasing schedule during each period. The initializing account can also choose whether to include such features as ability to revoke or pause within the contract.

Once these features are set the first time, they cannot be reset. Note that the account initializing the contract may not necessarily be the the `owner` of the contract instance.

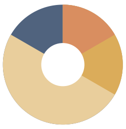
The total time of the vesting schedule is given by `period * vestedPerPeriod.length` , where `period` is the amount of time in seconds of each period, and `vestedPerPeriod.length` is the total number of periods. The total sum to be vested to the beneficiary is the summation of all the entries of array `vestedPerPeriod` .

If the vesting schedule is specified to be linear, then, in each period, the amount vested within that period can be released proportional to the amount of time spent in that period. Otherwise, the funds for the period only release after the period is completed.

The specified `owner` of the contract may `pause()` and `unpause()` the contract only if the contract `isPausable` (and if the contract is not yet revoked). The function `pause()` is designed to set the variable `stop` to the current `block.timestamp` . This is done so that, when paused, a beneficiary is not being vested after the timestamp stored as `stop` . This is a reversible action, so that if the owner decides, they can then call `unpause()` , which will set the variable `stop` back to 0. Doing this then allows the amount vested to continue to be calculated as if the contract was never paused.

The function `revoke()` is similar to the `pause()` function, except that it can only be executed once and it cannot be reversed. Moreover, it affects the amount available to be vested to the `beneficiary` . When called, any remaining non-vested amount out of the total amount to be vested is counted as surplus, allowing it to be released from the contract by the owner using function `releaseSurplus()` . As of the update in commit c8049ad98b543196647b2418dcca16cd3624009a, the function `revoke()` only updates `stop` if the contract is not currently paused (i.e., only if `stop` is currently zero). This accounts for the possibility in which the contract is first paused, time passes, and then the `owner` later decides function `revoke()` must be called. In such a scenario, `stop` should not be updated to a more recent timestamp, it should remain at the timestamp it was updated to when `pause()` was last called. The change in this commit accounts for that consideration.

FINDINGS | DECENTRALAND - PERIODICTOKENVESTING



6	0	1	1	3	1
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Decentraland - PeriodicTokenVesting. Through this audit, we have uncovered 6 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
PTV-01	Revoking Extends Value Of `stop` When Paused	Volatile Code	Medium	● Resolved
VES-01	**Centralization Risks In PeriodicTokenVesting.Sol**	**Centralization / Privilege**	**Major**	● **Acknowledged**
VES-02	Missing Checks	Volatile Code	Minor	● Acknowledged
VES-03	Third Party Dependency	Volatile Code	Minor	● Acknowledged
VES-04	Missing Address Validation	Volatile Code	Minor	● Acknowledged
PTV-02	Solidity Version Issues	Language Specific	Informational	● Resolved

PTV-01 | REVOKING EXTENDS VALUE OF `stop` WHEN PAUSED

Category	Severity	Location	Status
Volatile Code	● Medium	contracts/PeriodicTokenVesting.sol (base): 324	● Resolved

Description

In the `PeriodicTokenVesting` contract, the function `pause()` is designed to set the variable `stop` to the current `block.timestamp`. This is done so that when paused, a beneficiary is not being vested after the timestamp stored in `stop`. This is a reversible action, so that if the owner decides, they can then call `unpause()`, which will set the variable `stop` back to 0. Doing this then calculates the vested amount as if it was never paused.

The function `revoke()` is similar to the `pause()` function. The only difference being that `revoke()` is non-reversible and counts the remaining non-vested amount as surplus, allowing it to be released from the contract by the owner using `releaseSurplus()`. The issue here is that both of these set the variable `stop` to the current `block.timestamp` when they are called.

Impact

The following situation could arise. The owner pauses the contract setting `stop` to be the current `block.timestamp`. A significant amount of time then passes and the owner decides that they should `revoke()` as it is determined for certain that the `beneficiary` should no longer be entitled to vestings and that the unvested tokens should be counted as surplus. The owner in this case does not want to vest any additional tokens to the `beneficiary`. However, if the owner calls `revoke()`, then the variable `stop` will be adjusted to the current `block.timestamp`, which will then give the `beneficiary` additional vested tokens provided the new timestamp is after a `period/cliff` or the vesting is linear. In other words, the beneficiary will be vested until the point when `revoke()` was called, instead of when `pause()` was called.

Recommendation

We recommend updating the `stop` variable in the `revoke()` function only when the contract is not paused.

Alleviation

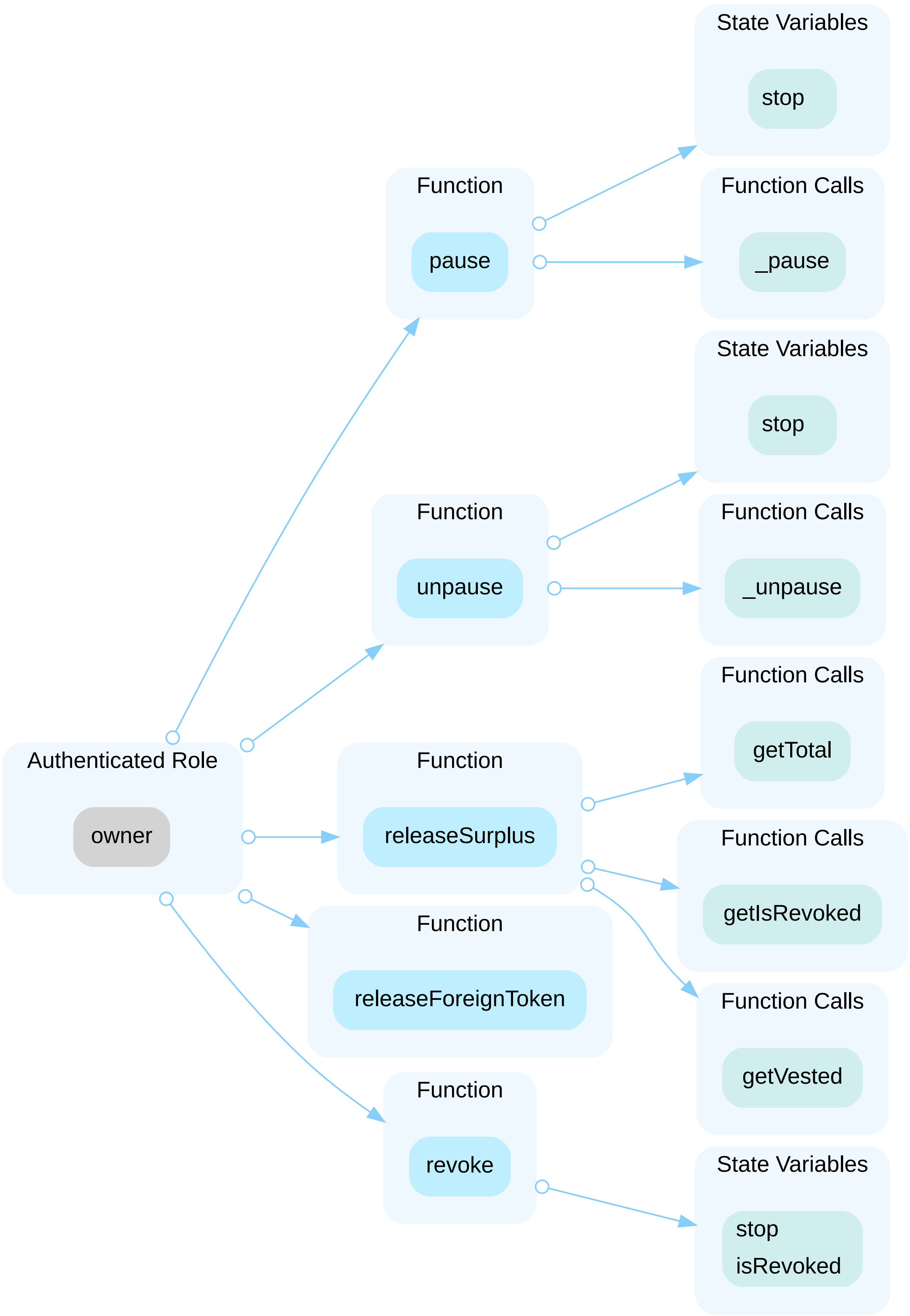
`[CertiK]`: The client resolved the issue.

VES-01 | CENTRALIZATION RISKS IN PERIODICTOKENVESTING.SOL

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	**contracts/PeriodicTokenVesting.sol (update2): 327~328, 344~345, 377~378, 432~433, 441~442; contracts/PeriodicTokenVesting.sol (base): 324, 338, 371, 426, 435**	● **Acknowledged**

Description

In the contract `PeriodicTokenVesting` the role `owner` has authority over the functions shown in the diagram below. Any compromise to the `owner` account may allow the hacker to take advantage of this authority and pause or unpause the contract state, revoke the vesting, withdraw any surplus token that is not needed for vesting through `releaseSurplus()`, and withdraw any other ERC20 tokens other than the vesting token through `releaseForeignToken()`.


State Variables
stop
Function
pause
Function Calls
_pause
State Variables
stop
Function
unpause
Function Calls
_unpause
Function Calls
getTotal
Authenticated Role
owner
Function
releaseSurplus
Function Calls
getIsRevoked
Function
releaseForeignToken
Function Calls
getVested
Function
revoke
State Variables
stop
isRevoked

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

`[CertiK]` : The client acknowledged the issue.

`[Decentraland]` : "A MultiSig will be used as owner of the contract for our own use cases."

VES-02 | MISSING CHECKS

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/PeriodicTokenVesting.sol (update2): 104~105, 138~139; contracts/PeriodicTokenVesting.sol (base): 104~105, 136	● Acknowledged

Description

In the `initialize()` function, it is not checked if the value of cliff is less than the total vesting duration.

Additionally, there is no check that the uint entries of array `_vestedPerPeriod` are nonzero values. This allows for vestment periods in which the `beneficiary` address will not acquire any newly releasable tokens.

Recommendation

We recommend adding a check that input `_vestedPerPeriod` only contains nonzero uint values and adding a check that the cliff value is less than the total vesting duration.

Alleviation

`[CertiK]` : The client acknowledged the issue.

VES-03 | THIRD PARTY DEPENDENCY

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/PeriodicTokenVesting.sol (update2): 18~19; contracts/PeriodicTokenVesting.sol (base): 18~19	● Acknowledged

Description

The contract is serving as the underlying entity to interact with a third party token. The scope of the audit treats 3rd party entities as black boxes and assumes their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. Additionally, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic of `PeriodicTokenVesting` may require interaction with a third party ERC20 token. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

`[Decentraland]` : "Issue acknowledged. Won't make any changes for the current version."

VES-04 | MISSING ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/PeriodicTokenVesting.sol (update2): 95~96, 96~97; contracts/PeriodicTokenVesting.sol (base): 95~96, 96~97	● Acknowledged

Description

The function `initialize()` is missing a check that addresses `_owner` and `_beneficiary` are not the same address.

If the `_owner` and `_beneficiary` are the same address, then the `beneficiary` of the contract instance can call `revoke()` and `redeemSurplus()` (as the `owner`) in sequence to immediately acquire a majority, if not all, of the contract's balance of `token` .

Recommendation

We recommend adding the check outlined above.

Alleviation

[`Decentraland`]: "Issue acknowledged. Won't make any changes for the current version."

PTV-02 | SOLIDITY VERSION ISSUES

Category	Severity	Location	Status
Language Specific	● Informational	contracts/PeriodicTokenVesting.sol (base): 3	● Resolved

Description

The contract uses the latest solidity version available at the moment which is `0.8.17`. Using the most recent version can be risky as all bugs for the current version may not be known.

Furthermore, the contract has an unlocked compiler version. An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to ambiguity when debugging, as compiler-specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We recommend deploying the contract with a less recent version of Solidity (the contract can compile from 0.8.2 and above) and locking the version.

Alleviation

`[CertiK]`: The client made the recommended changes.

APPENDIX | DECENTRALAND - PERIODICTOKENVESTING

Finding Categories

Categories	Description
Centralization / Privilege	Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.
Language Specific	Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you (“Customer” or the “Company”) in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK’s prior written consent in each instance.

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. CertiK’s position is that each company and individual are responsible for their own due diligence and continuous security. CertiK’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED “AS IS” AND “AS AVAILABLE” AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER’S OR ANY OTHER PERSON’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE

FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK’S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER’S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED “AS IS” AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK’S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we’re able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.