# Preliminary Comments

# VCGamers

Feb 11th, 2022

# Table of Contents

# Summary

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

- 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.

# Overview

## Project Summary

| | |
|---|---|
| Project Name | VCGamers |
| Description | ERC20 and Tax Collector |
| Platform | bsc |
| Language | Solidity |
| Codebase | https://github.com/vcgamers/smartcontract/tree/5e0cdb7dcdc23b967c12d0413830b81261022bfb |
| Commit | |

## Audit Summary

| | |
|---|---|
| Delivery Date | Feb 11, 2022 |
| Audit Methodology | Static Analysis, Manual Review |

## Vulnerability Summary

| Vulnerability Level | Total | Pending | Declined | Acknowledged | Partially Resolved | Mitigated | Resolved |
|---|---|---|---|---|---|---|---|
| Critical | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Major | 3 | 0 | 0 | 3 | 0 | 0 | 0 |
| Medium | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| Minor | 2 | 0 | 0 | 2 | 0 | 0 | 0 |
| Informational | 7 | 0 | 0 | 7 | 0 | 0 | 0 |
| Discussion | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## Audit Scope

| ID | File | SHA256 Checksum |
| --- | --- | --- |
| VCG | VCG | 2ecf4df3471ba42c8c114b42f4a00d310275e65b53f4d3c3ddfa5f4da24f2e67 |

# Findings


| | |
|---|---|
| Critical | 0 (0.00%) |
| Major | 3 (23.08%) |
| Medium | 1 (7.69%) |
| Minor | 2 (15.38%) |
| Informational | 7 (53.85%) |
| Discussion | 0 (0.00%) |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| VCG-01 | Centralization Related Risks | Centralization / Privilege | Major | Acknowledged |
| VCG-02 | No Upper Limit For Fees | Control Flow | Major | Acknowledged |
| VCG-03 | Initial Token Distribution | Centralization / Privilege | Major | Acknowledged |
| VCG-04 | `VCG` Missing `iTaxCollector` Ownership | Logical Issue | Medium | Acknowledged |
| VCG-05 | Check-effect-interaction Pattern Violation | Logical Issue | Minor | Acknowledged |
| VCG-06 | `swapping()` Modifier Not Utilized | Logical Issue | Minor | Acknowledged |
| VCG-07 | Missing Emit Events | Coding Style | Informational | Acknowledged |
| VCG-08 | Unlocked Compiler Version | Language Specific | Informational | Acknowledged |
| VCG-09 | Declare Variable as `constant` | Gas Optimization | Informational | Acknowledged |
| VCG-10 | Unused Variables | Gas Optimization | Informational | Acknowledged |
| VCG-11 | Interface Not Inherited | Coding Style | Informational | Acknowledged |
| VCG-12 | Uncallable Function | Volatile Code | Informational | Acknowledged |
| VCG-13 | SafeMath Included in Pragma | Language Specific | Informational | Acknowledged |

## VCG-01 | Centralization Related Risks

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | Major | VCG: 59, 63, 213, 216, 219, 224, 239, 266, 267, 268, 287, 288, 316, 317, 318, 319, 320, 321 | Acknowledged |

### Description

In the contract `VCG`, the role `_owner` has authority over the following functions:

- `setTax()`
- `setTaxDivider()`
- `setTaxCollector()`
- `changeExcludeBuyFee()`
- `changeExcludeSellFee()`
- `taxBalance()`
- `distributeTax()`
- `setTeamTax()`
- `setPartnershipTax()`
- `setRNDTax()`
- `setTaxTransferOwner()`

In the contract `TaxCollector`, the role `_owner` has authority over the following functions:

- `setTeam()`
- `setPartnership()`
- `setRND()`
- `distribute()`
- `kill()`

Any compromise to the `_owner` account may allow a hacker to take advantage of this authority. For example, the hacker could call `setTaxTransferOwner()` function and set the owner of the `TaxCollector` contract to an hacker affiliated address, and then call the `kill()` function in `TaxCollector` to drain all token balance from `TaxCollector` and delete the contract.

### 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., multi-signature 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 (⅗, ⅓) 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.

*Noted: Recommend considering the long-term solution or the permanent solution. The project team shall make a decision based on the current state of their project, timeline, and project resources.*

## Alleviation

***[VCGamers Team]***: We have created a multisign wallet for avoiding the hacker of the centralized privilege functions. GnosisSafeProxy
https://bscscan.com/address/0xd9cEec57e119d65Fb064578DD1E35B3bDB661878

## VCG-02 | No Upper Limit For Fees

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Control Flow | Major | VCG: 266, 267 | Acknowledged |

### Description

The `_owner` can set the `_tax` and `_taxDivider` values in the contract and there is no upper limit on what the rate can be. In the extreme case, the rate can be as high as 100%, which would imply that users cannot buy or sell the token.

### Recommendation

We recommend setting a reasonable upper limit for tax related variables.

### Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-03 | Initial Token Distribution

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | Major | VCG: 303 | Acknowledged |

### Description

All of the VCG tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the deployer can distribute VCG tokens without obtaining the consensus of the community.

### Recommendation

We recommend the team to be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.

### Alleviation

*[VCGamers Team]*: OurInitial Token Distribution was for Private and Pre-sale. Presale contract : 0xEd7e7AA86eba84fA301a331bF9Db9e3ad176302A

## VCG-04 | `VCG` Missing `iTaxCollector` Ownership

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Logical Issue | Medium | VCG: 271 | Acknowledged |

### Description

There is no check to ensure that the contract `VCG` will properly be able to call the owner privileged functions of `iTaxCollector` as there is no check to guarantee that ownership has been transferred.

### Recommendation

We recommend instantiating the `iTaxCollector` in the constructor of the contact, limiting it within the scope of `VCG`.

### Alleviation

***[VCGamers Team]***: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-05 | Check-effect-interaction Pattern Violation

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Logical Issue | Minor | VCG: 375 | Acknowledged |

### Description

In the function `_transferTax()`, the value of `_balances[recipient]` is updated after the function `distributeFee()` (makes external call) is called when selling. This violates the check-effect-interaction pattern.

### Recommendation

We recommend using the Checks-Effects-Interactions Pattern to avoid the risk of calling unknown contracts or applying OpenZeppelin ReentrancyGuard library - `nonReentrant` modifier for the aforementioned functions to prevent reentrancy attack.

### Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-06 | `swapping()` Modifier Not Utilized

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Logical Issue | Minor | VCG: 208, 224~237 | Acknowledged |

### Description

In the `TaxCollector` contract, the `swapping()` modifier doesn't contain any `require` statement and merely updates the `inSwap` boolean variable, which is not utilized in the `TaxCollector` contract. As a result, adding the modifier `swapping()` to the `distribute()` function doesn't prevent reentrancy or have any other effect.

### Recommendation

We recommend adding a `require` statement to the `swapping()` modifier if the intention is to prevent Reentrancy in the `distribute()` function.

### Alleviation

There is no response from the client regarding this finding.

## VCG-07 | Missing Emit Events

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Coding Style | Informational | VCG: 213, 216, 219, 224, 239, 266~268, 287~288, 317~321 | Acknowledged |

### Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

### Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

### Alleviation

There is no response from the client regarding this finding.

## VCG-08 | Unlocked Compiler Version

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Language Specific | Informational | VCG: 16 | Acknowledged |

### Description

The contract has 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 different compiler versions. This can lead to an 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 advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at. For example, for version `v0.8.4` the contract should contain the following line:

```
pragma solidity 0.8.4;
```

### Alleviation

There is no response from the client regarding this finding.

## VCG-09 | Declare Variable As `constant`

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Gas Optimization | Informational | VCG: 210, 274, 275, 276, 277, 278, 261, 259, 260 | Acknowledged |

### Description

Variables `maxPercent`, `_name`, `_symbol`, `_decimals`, `DEAD`, `ZERO`, `ROUTER`, `FACTORY`, and `WBNB` could be declared as `constant` since these state variables are never to be changed.

### Recommendation

We recommend declaring those variables as `constant`.

### Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-10 | Unused Variables

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Gas Optimization | Informational | VCG: 274, 275 | Acknowledged |

### Description

The state variables `ZERO` and `DEAD` are never used nor changed.

### Recommendation

We advise removing the unused state variables.

### Alleviation

There is no response from the client regarding this finding.

## VCG-11 | Interface Not Inherited

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Coding Style | Informational | VCG: 208 | Acknowledged |

### Description

The interface `ITaxCollector` is present but not inherited by `TaxCollector`.

### Recommendation

We recommend having `TaxCollector` inherit `ITaxCollector`.

### Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-12 | Uncallable Function

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Volatile Code | Informational | VCG: 441 | Acknowledged |

### Description

The function `_burn()` is visibly `internal` however there is no `external` or `public` function that call `_burn()`. Therefore `_burn()` is unable to be called.

### Recommendation

We recommend removing the unused functionality.

### Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

## VCG-13 | SafeMath Included In Pragma

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Language Specific | Informational | VCG: 187 | Acknowledged |

## Description

Solidity version >=0.8.0 includes checked arithmetic operations and underflow/overflow by default, making SafeMath redundant.

## Recommendation

We recommend removing the SafeMath library and use standard arithmetic operators to reduce code complexity.

## Alleviation

*[VCGamers Team]*: we will do in the future, when we do the migrating all of the issue will be resolved. but for now we are focussing on existing souce code

# Appendix

## Finding Categories

### 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.

### Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

### Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

### Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

### 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.

### Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

## 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

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.

# About

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.