CERTIK

# Security Assessment

## GrowingFi

Jul 8th, 2021

# Table of Contents

## Summary

This report has been prepared for GrowingFi to discover issues and vulnerabilities in the source code of the GrowingFi 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 given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

# Overview

## Project Summary

| | |
|---|---|
| Project Name | GrowingFi |
| Description | Growing Farms is a yield optimizer platform on BSC, focusing on providing auto-compounded yields by actively finding, auditing, and leveraging the best yield farming platforms through smart optimizing strategies. |
| Platform | BSC |
| Language | Solidity |
| Codebase | https://github.com/growingfi/contracts/ |
| Commit | 1ac16d9bded45b144add002c3447d06482405fc0 |

## Audit Summary

| | |
|---|---|
| Delivery Date | Jul 08, 2021 |
| Audit Methodology | Static Analysis, Manual Review |
| Key Components | Strategie, Staking |

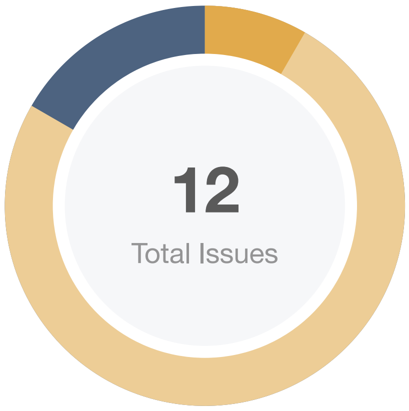
## Vulnerability Summary

| Vulnerability Level | Total | Pending | Partially Resolved | Resolved | Acknowledged | Declined |
|---|---|---|---|---|---|---|
| ● Critical | 0 | 0 | 0 | 0 | 0 | 0 |
| ● Major | 0 | 0 | 0 | 0 | 0 | 0 |
| ● Medium | 1 | 0 | 0 | 1 | 0 | 0 |
| ● Minor | 9 | 0 | 0 | 0 | 9 | 0 |
| ● Informational | 2 | 0 | 0 | 0 | 2 | 0 |
| ● Discussion | 0 | 0 | 0 | 0 | 0 | 0 |

## Audit Scope

| ID | file | SHA256 Checksum |
|---|---|---|

## Findings



| | |
|---|---|
| 🟥 **Critical** | **0** (0.00%) |
| 🟧 **Major** | **0** (0.00%) |
| 🟨 **Medium** | **1** (8.33%) |
| 🟨 **Minor** | **9** (75.00%) |
| 🟦 **Informational** | **2** (16.67%) |
| 🟩 **Discussion** | **0** (0.00%) |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| GLOBAL-01 | Third Party Dependencies | Volatile Code | ● Minor | ⓘ Acknowledged |
| **GLOBAL-02** | Privileged roles | **Centralization / Privilege** | ● **Minor** | ⓘ **Acknowledged** |
| BGS-01 | Non-optimal conditional statement | Gas Optimization | ● Informational | ⓘ Acknowledged |
| BGS-02 | Return value not handled | Volatile Code | ● Minor | ⓘ Acknowledged |
| GMI-01 | Lack of zero address check | Volatile Code | ● Minor | ⓘ Acknowledged |
| GMI-02 | Missing Emit Events | Coding Style | ● Informational | ⓘ Acknowledged |
| GMI-03 | Incorrect error message | Logical Issue | ● Minor | ⓘ Acknowledged |
| GRI-01 | Lack of zero address check | Volatile Code | ● Minor | ⓘ Acknowledged |
| GSL-01 | Lack of zero address check | Volatile Code | ● Minor | ⓘ Acknowledged |
| GSP-01 | Lack of zero address check | Volatile Code | ● Minor | ⓘ Acknowledged |
| SUI-01 | Lack of zero address check | Volatile Code | ● Minor | ⓘ Acknowledged |
| SUI-02 | Potential Flashloan Attack | Volatile Code | ● Medium | ✓ Resolved |

## GLOBAL-01 | Third Party Dependencies

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | Global | ⓘ Acknowledged |

### Description

The contract is serving as the underlying entity to interact with third party protocols such as Pancakeswap, AlpacaFinance, Autofarm and SwampFinance. 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. In addition, 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 Growingfi requires interaction with third party dependencies. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

## GLOBAL-02 | Privileged roles

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | ● Minor | Global | ⓘ Acknowledged |

### Description

There is one noticeable high privilege role across different contracts in the GrowingFi project. The owner can modify critical configurations for one or more contracts. If an attacker takes control over any one of the roles, the actions he can perform will endanger the users' farming reward and the value of the Grow token.

The overly powerful owner is a centralization risk as he can perform actions without obtaining the consensus of the community.

### Recommendation

We advise the client to handle the privileged role carefully to avoid any potential hack. In addition, we advise the client to consider the following solutions:

1. `Timelock` with reasonable latency for community awareness on privileged operations;
2. Multisig with community-voted 3rd-party independent co-signers;
3. DAO or Governance module increasing transparency and community involvement.

## BGS-01 | Non-optimal conditional statement

| Category | Severity | Location | Status |
|---|---|---|---|
| Gas Optimization | ● Informational | contracts/strategies/BaseGrowStrategy.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 234~248 | ⓘ Acknowledged |

### Description

The value of 'shareRemoved' is first calculated by taking the minimum of 'userShares[msg.sender]' and a second value that requires a number of calculations. The value of 'shareRemoved' is reverted to userShares[msg.sender] if an inequality is satisfied. In the current order, both the minimum operation and inequality are executed, and these operations cost gas.

### Recommendation

It is recommended that the client consider using the if..then construct to first check the inequality and, if not satisfied, perform the minimum of the two values.

## BGS-02 | Return value not handled

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/strategies/BaseGrowStrategy.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 81~93 | ⓘ Acknowledged |

### Description

Within BaseGrowStrategy.sol, `IERC20(tokenB).transerFrom(..)` performs an external call to `transferFrom()` and the return value is not checked. There is a danger in continuing the execution of code without considering the returned value in the workflow especially in the edge cases when there are insufficient funds or network communication errors.

### Recommendation

It is recommended to use SafeERC20 or make sure that the value returned from 'transferFrom()' is checked.

## GMI-01 | Lack of zero address check

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/grow/GrowMinter.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 79 | ⓘ Acknowledged |

### Description

Across multiple contracts of the project, there is a lack of zero address validation performed in functions that involve the transfer of funds or an update to privileged roles. It is important to check for the zero address, otherwise funds irrevocably lost or contract ownership could be unintentionally altered. In the case of a zero address used within an update to privileged roles, essential functionality of the contract could be removed.

### Recommendation

It is recommended to check that the target address is not the zero address and handle the case of zero address with the appropriate error message.

## GMI-02 | Missing Emit Events

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | ● Informational | contracts/grow/GrowMinter.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 90 | ⓘ Acknowledged |

### Description

In the function `mint()`, code to emit the event 'LogGrowMint()' defined in GrowReward.sol is commented out but could be included to improve the clarity within the log and provide the expected transparency of the contract.

### Recommendation

It is recommended that the client consider adding events for sensitive actions and emit them in the function following the Checks-Effects-Interactions best practices of Solidity programming.

## GMI-03 | Incorrect error message

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | contracts/grow/GrowMinter.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 185 | ⓘ Acknowledged |

### Description

A `require` statement is used to check if the `strategyAddress` exists in the strategies array. The condition will fail if the strategy doesn't exist, and the current error message in the code is "GrowMaster: strategy is already set." We believe the error message should be "GrowMaster: strategy doesn't exist".

### Recommendation

It is recommended that the client update the error message accordingly.

## GRI-01 | Lack of zero address check

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/grow/GrowRewarder.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 64~67, 74~77 | ⓘ Acknowledged |

### Description

Across multiple contracts of the project, there is a lack of zero address validation performed in functions that involve the transfer of funds or an update to privileged roles. It is important to check for the zero address, otherwise funds irrevocably lost or contract ownership could be unintentionally altered. In the case of a zero address used within an update to privileged roles, essential functionality of the contract could be removed.

### Recommendation

It is recommended to check that the target address is not the zero address and handle the case of zero address with the appropriate error message.

## GSL-01 | Lack of zero address check

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/strategies/GrowStrategyAutoLike.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 47 | ⓘ Acknowledged |

### Description

Across multiple contracts of the project, there is a lack of zero address validation performed in functions that involve the transfer of funds or an update to privileged roles. It is important to check for the zero address, otherwise funds irrevocably lost or contract ownership could be unintentionally altered. In the case of a zero address used within an update to privileged roles, essential functionality of the contract could be removed.

### Recommendation

It is recommended to check that the target address is not the zero address and handle the case of zero address with the appropriate error message.

## GSP-01 | Lack of zero address check

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/grow/GrowStakingPool.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 45, 71 | ⓘ Acknowledged |

### Description

Across multiple contracts of the project, there is a lack of zero address validation performed in functions that involve the transfer of funds or an update to privileged roles. It is important to check for the zero address, otherwise funds irrevocably lost or contract ownership could be unintentionally altered. In the case of a zero address used within an update to privileged roles, essential functionality of the contract could be removed.

### Recommendation

It is recommended to check that the target address is not the zero address and handle the case of zero address with the appropriate error message.

## SUI-01 | Lack of zero address check

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Minor | contracts/utils/SwapUtils.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 52~54 | ⓘ Acknowledged |

### Description

Across multiple contracts of the project, there is a lack of zero address validation performed in functions that involve the transfer of funds or an update to privileged roles. It is important to check for the zero address, otherwise funds irrevocably lost or contract ownership could be unintentionally altered. In the case of a zero address used within an update to privileged roles, essential functionality of the contract could be removed.

### Recommendation

It is recommended to check that the target address is not the zero address and handle the case of zero address with the appropriate error message.

## SUI-02 | Potential Flashloan Attack

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | ● Medium | contracts/utils/SwapUtils.sol (400fcd0d50a49bbbb2f645efb93af1d152d597d2): 53 | ✓ Resolved |

### Description

Flash loans are a way to borrow large amounts of money for a certain fee. The requirement is that the loans need to be returned within the same transaction in a block. If not, the transaction will be reverted.

An attacker can use the borrowed money as the initial funds for an exploit to enlarge the profit and/or manipulate the token price in the decentralized exchanges.

We find that the `tokenPriceInBNB` function relies on price calculations that are based on-chain, meaning that they would be susceptible to flash-loan attacks by manipulating the price of given pairs to the attacker's benefit.

### Recommendation

If a project requires price references, it needs to be careful of flash loans that might manipulate token prices. To prevent this from happening, we recommend the following:

1. Use a reliable on-chain price oracle, such as Chainlink.
2. Use Time-Weighted Average Price (TWAP). The TWAP represents the average price of a token over a specified time frame. If an attacker manipulates the price in one block, it will not affect the average price too much.

### Alleviation

The updated `tokenPriceInBNB` function now uses ChainLink as the price oracle and TWAP price.

# 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.

### Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

### 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 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 without CertiK's prior written consent.

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.

## 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.

CERTIK