DeHacker

Code Security Assessment

# Opulous (Staking Pools)

March5th, 2023

# Contents

# Summary

DeHacker's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.
Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow/underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service/logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

# Issue Categories

Every issue in this report was assigned a severity level from the following:

### Critical severity issues

A vulnerability that can disrupt the contract functioning in a number of scenarios or creates a risk that the contract may be broken.

### Major severity issues

A vulnerability that affects the desired outcome when using a contract or provides the opportunity to use a contract in an unintended way.

### Medium severity issues

A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.

### Minor severity issues

A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.

### Informational

A vulnerability that has informational character but is not affecting any of the code.

# Overview

## Project Summary

| | |
|---|---|
| **Project Name** | Opulous (Staking Pools) Addendum |
| **Platform** | Algorand (ALGO) |
| **website** | https://opulous.org/ |
| **Type** | Staking |
| **Language** | Solidity |

## Vulnerability Summary

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

# Audit scope

| ID | File | SHA256 Checksum |
|---|---|---|
| OAK | projects/Opulous Addendum/contracts/staking-pool-two-tokens-approval.teal | a3d46b6225eb6248a25c0cbc527bc92b288af4a0d4c057a470be11b84c7daf59 |
| OAP | projects/Opulous Addendum/contracts/staking-pool-two-tokens-clear.teal | 5b33c90270f46f80893c4f52e5b6b5a44f0576952feb58a6f5fbd85427ddb667 |

# Findings

| ID | Category | Severity | Status |
|---|---|---|---|
| GLOBAL-01 | Centralization /Privilege | Major | Acknowledged |
| GLOBAL-02 | Centralization /Privilege | Major | Acknowledged |
| GLOBAL-03 | Control Flow | Medium | Resolved |
| GLOBAL-04 | Business Model | Informational | Resolved |

# Major

## GLOBAL-01 | CENTRALIZATION RELATED RISKS

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | Major | | Acknowledged |

## Description

In the contract staking-pool-two-tokens-approval , the role admin has authority over the following functions:

setup()
add_whitelist()
remove_whitelist()
close()
set_admin_address()
set_max_pool_size()
set_max_share()
set_end_timestamp()
set_lockup_and_apy()
defund()

Any compromise to the admin account may allow a hacker to take advantage of this authority.

## Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level ofdecentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefullymanage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommendcentralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accountswith enhanced security practices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at different levels:

Multi sign (2/3, 3/5) mitigate and avoids a single point of key management failure.

Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private keybeing compromised;
AND
A medium/blog link for sharing the multi-signers addresses 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: The project team shall make a decision based on the current state of their project , timeline, and project resources.

# Major

## GLOBAL-02 | DEPENDENCY ON FOREIGN ASSETS

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | Major | | Acknowledged |

### Description

The scope of this project deals with the transfer of foreign assets, e.g: OPUL, MFT & USDC as either stake tokens or rewardtokens. Algorand supports asset freezing and clawback for tokens, which might pose a risk to this program.

### Recommendation

We encourage the team to monitor the deployment of stake and reward tokens to make sure the pooled assets can not befreezed or clawbacked

# Medium

## GLOBAL-03 | MISSING TXN BASIC CHECKS

| Category | Severity | Location | Status |
|---|---|---|---|
| Control Flow | Medium | | Resolved |

### Description

The smart contract is missing some significant checks on transactions.

### Recommendation

We advise the team to:
1. Verify that the RekeyTo field is set to the ZeroAddress unless the contract is specifically involved in a rekeyingoperation
2. Verify that CloseRemainderTo and AssetCloseTo is set to the ZeroAddress if without an intended recipient
3. Verify that AssetSender is set to the ZeroAddress to avoid a clawback transaction
Reference: https://developer.algorand.org/docs/get-details/dapps/avm/teal/guidelines/ andhttps://developer.algorand.org/docs/get-details/transactions/transactions/#asset-transfer-transaction

# Informational

## GLOBAL-04 | WHISTLIST TOKENS CAN BE TRANSFERRED BY THEFIRST WHITELISTED ADDRESSES

| Category | Severity | Location | Status |
|---|---|---|---|
| Business Model | Informational | | Resolved |

## Description

According to the Opulous staking pool - Two Tokens Documentation, whitelist tokens will be used in the pool to control useraccess, and users who have a balance of the token will be able to stake in the pool.
Our current understanding is that if the target ASA is owned by an admin address, then the admin can determine the initialset of addresses by transferring the target ASA to them. For example, let's say each address receives 10 tokenAlpha fromthe admin. Then the addresses with tokenAlpha can transfer their tokenAlpha balance to more addresses. Then all theseaddresses with tokenAlpha balance have permission to stake to the pool. We would like to confirm if the above-describedbehavior is intended and allowed.
At meanwhile, it seems the add_whitelist branch can be renamed to set_whitelist , since it does not check if thewhitelist_token_id is empty before app_global_put . We would like to confirm the intention of this branch as well

## Recommendation

There are no security concern for this finding. If the above-described behavior is not allowed, some possible approachescould be like increasing the minimum balance threshold in check_is_whitelisted , or like only distributing the minimumbalance to the initial set of whitelisted addresses, etc

# Optimizations

| ID | Category | Severity | Status |
| --- | --- | --- | --- |
| OAK-01 | Mathematical Operati | Optimization | Resolved |

# Optimization

## OAK-01 | EXTRACTING stake:amount

| Category | Severity | Location | Status |
|---|---|---|---|
| Mathematical Operations | Optimization | projects/Opulous Addendum/ contracts/staking-pool-two-tokens-approval.teal: 1270 | Resolved |

## Description

stake:amount is stored in the local mapping with stake:slot to be the key, and it is read and copied to the slot 2 of thescratchspace each time when the subroutine set_stake_props is triggered. For the other three fields in the mapping,slot:time , slot:lockup and slot:apy , they are not likely to reach the max value if the inputs are valid. The opcodeextract_uint64 will throw an error and revert the transaction if the number to be casted is greater than the max value ofuint64.

## Recommendation

We recommend the team to check all related parameters in setup and always makes sure that uint128(stake:amount *stake_apy * rewarded time) / uint128(seconds in year * 10000) is a pure uint64 number less than uint64.max =18446744073709551615 , or use uint128 for all calculations

# Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

# 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.

### Coding Style

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

### Volatile Code

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

### 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.

## 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.

# About

DeHacker is a team of auditors and white hat hackers who perform security audits and assessments. With decades of experience in security and distributed systems, our experts focus on the ins and outs of system security. Our services follow clear and prudent industry standards. Whether it's reviewing the smallest modifications or a new platform, we'll provide an in-depth security survey at every stage of your company's project. We provide comprehensive vulnerability reports and identify structural inefficiencies in smart contract code, combining high-end security research with a real-world attacker mindset to reduce risk and harden code.

### BLOCKCHAIINS

Ethereum
Cosmos
Eos
Substrate

### TECH STACK

Python
Solidity
Rust
c++

### CONTACTS

https://dehacker.io
https://twitter.com/dehackerio
https://github.com/dehacker/audits_public
https://t.me/dehackerio
https://blog.dehacker.io/

DeHacker

March 2023