Security Assessment

PandaSwap

May 3rd, 2021

Summary

This report has been prepared for PandaSwap smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Dynamic Analysis, 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	PandaSwap
Description	PandaSwap is a decentralized exchange running on OKEx Chain, with lots of other features that let you earn and win tokens.
Platform	OKExChain
Language	Solidity
Codebase	https://www.oklink.com/okexchain-test/address/0xc0ea1b065d268e71d71f9f6c6ba93f4bbca5e31f
Commits	1. https://www.oklink.com/okexchain-test/address/0xc0ea1b065d268e71d71f9f6c6ba93f4bbca5e31f 2. sha256 of file: bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb

Audit Summary

Delivery Date	May 03, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

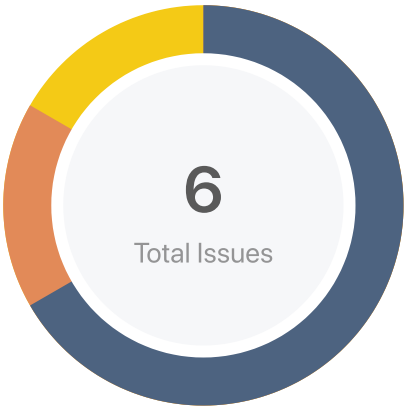
Total Issues	6
Critical	0
Major	1
Medium	0
Minor	1
Informational	4

Discussion	0

Audit Scope

ID	file	SHA256 Checksum
CKP	main.sol	735f8460e1d9e0446adc5f3cbf7593f09f040b8580ac62e7eeeeaacf586e9619

Findings



Critical	0 (0.00%)
Major	1 (16.67%)
Medium	0 (0.00%)
Minor	1 (16.67%)
Informational	4 (66.67%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
CKP-01	SafeMath Not Used	Mathematical Operations	Informational	Resolved
CKP-02	Lack of Input Validation	Volatile Code	Informational	Resolved
CKP-03	Sold Amount Not Check	Logical Issue	Minor	Resolved
CKP-04	Administrator Capability	Logical Issue	Major	Acknowledged
CKP-05	Compares to a Boolean Constant	Optimization	Informational	Resolved
CKP-06	Redundant Comparison	Optimization	Informational	Resolved

CKP-01 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Informational	main.sol: 196, 202	Resolved

Description

SafeMath is not used making it possible for overflow/underflow, which will lead to an inaccurate message.

Recommendation

Consider using SafeMath library

```
addressBoughtMap[address(msg.sender)].add(amount)
```

Alleviation

The development team heeded our advice and resolved this issue in source file whose shasum 256 is bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb.

CKP-02 | Lack of Input Validation

Category	Severity	Location	Status
Volatile Code	Informational	main.sol: 164, 167, 217	Resolved

Description

The assigned value to `saleToken` in the constructor of `IDOOversubscribe` should be verified as a non-zero address.

The assigned value to `saleTokenDecimals` and `reciveTokenDecimals` in the constructor of `IDOOversubscribe` should be verified as a non-zero value.

The assigned value to `minAmount` in function `start` should be verified as a non-zero value.

Recommendation

Check that the address is not zero by adding following checks.

```
require(_saleToken != address(0), "_saleToken is a zero address");
require(_saleTokenDecimals != 0, "_saleTokenDecimals is a zero value");
require(_reciveTokenDecimals != 0, "_reciveTokenDecimals is a zero value");
```

```
require(_minAmount != 0, "_minAmount is a zero value");
```

Alleviation

The development team heeded our advice and resolved this issue in source file whose shasum 256 is bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb.

CKP-03 | Sold Amount Not Check

Category	Severity	Location	Status
Logical Issue	Minor	main.sol: 203~205	Resolved

Description

When the IDO is not allowed to be overSubscribed, the sold amount should be checked.

Recommendation

Consider checking the sold amount as below:

```
if(!isOversubscribe) {
            require(sale.sold.add(bought)<=sale.amount,"over subscribe");
            IERC20(saleToken).safeTransfer(msg.sender, bought);
        }
```

Alleviation

The development team heeded our advice and resolved this issue in source file whose shasum 256 is bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb.

CKP-04 | Administrator Capability

Category	Severity	Location	Status
Logical Issue	Major	main.sol: 226, 240, 293	Acknowledged

Description

To bridge the trust gap between the administrator and users, the administrator needs to express a sincere attitude with the consideration of the administrator team's anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:

- Administrator can transfer tokens to any account under unpredicted cases via `withdraw`, `withdrawSaleToken`, and `withdrawReceiveToken` functions.

Recommendation

To improve the trustworthiness of the project, dynamic runtime changes on the protocol should be notified to clients. Any plan to call `withdraw`, `withdrawSaleToken` or `withdrawReceiveToken` function is better to move to the execution queue of Timelock.

Alleviation

The development team responded as below: According to the different requirements of the project initiators, currently, we need to keep the freedom on whether to use time lock for the funds raised. In the future, this will combine with our DAO.

CKP-05 | Compares to a Boolean Constant

Category	Severity	Location	Status
Optimization	Informational	main.sol: 249~250	Resolved

Description

Compares to a boolean constant.

```
396   require(isOversubscribe == true, 'not oversubscribe');
397   require(addressClaimedMap[address(msg.sender)] != true, 'already claimed');
```

Recommendation

Consider removing the equality to the boolean constant.

Alleviation

The development team heeded our advice and resolved this issue in source file whose shasum 256 is bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb.

CKP-06 | Redundant Comparison

Category	Severity	Location	Status
Optimization	Informational	main.sol: 192~193	Resolved

Description

Redundant comparison as below:

```
192  require(sale.startTime > 0 && now >= sale.startTime,'IDO not start');
193  require(now <= sale.closeTime,'IDO is over !');
```

since `sale.startTime` will be initialized in the function `start` to be greater than `now`. In other side, the comparison `now <= sale.closeTime` will return false if the sale is not initialized and still no need to compare `sale.startTime` to zero.

Recommendation

Consider refactoring codes as below:

```
require(now >= sale.startTime && now <= sale.closeTime,'IDO is not active');
```

Alleviation

The development team heeded our advice and resolved this issue in source file whose shasum 256 is bbfff639baa0b81333a88a5cc5d61d1903511f26598781072254e5fb6c9efacb.

Appendix

Finding Categories

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.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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.

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a struct assignment operation affecting an in-memory struct rather than an in-storage one.

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.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

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.