Guard

Smart contracts security assessment

Final report

Tariff: Standard

...

OpenSwap

September 2021

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for The Open Finance Project.

Code was audited after the 92071f2 commit.

Name	OpenSwap
Audit date	2021-09-24 - 2021-09-27
Language	Solidity
Platform	Harmony

Contracts checked

Name	Address
All contracts	
SushiToken	0xc0431Ddcc0D213Bf27EcEcA8C2362c0d0208c6DC
SushiMaker	0x7954dA0b4A81019A9313403FCDe072B93Aa41d36
Ownable	
TokenLock	0x8c4245b6096EE6e3C7266f4289233E93B24f0b2d
MasterChef	0xaC71B617a58B3CC136D1f6A118252f331faB44fC
UniswapV2ERC20	0xFdedbD274025675e17603c8A9a92Bfaa13e9d5c7
UniswapV2Router02	0xE6a72FeE7e34768661805DE2b621a8CDBe0DBc81

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools

- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Manually analyse smart contracts for security vulnerabilities
- Smart contracts' logic check

Known vulnerabilities checked

Title	Check result
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	not passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed
Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed

State Variable Default Visibility	passed
Reentrancy	not passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	passed
Outdated Compiler Version	not passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

1. No support of deflationary tokens (MasterChef)

If the owner adds pool with deflationary token, math of OpenSwap token distribution will break and users may not withdraw his funds.

Recommendation: Do not use deflationary tokens or check real amount of deposited tokens that were transferred to the contract by checking balances before and after token transfer.

2. Reentrancy in emergencyWithdraw (MasterChef)

In function emergencyWithdraw() L325 can be produced reentrancy attack if the token is bad. Because of that all pid can be withdrawn.

Recommendation: Add a reentrancy guard.

Medium severity issues

1. Delegates not transferred (SushiToken)

In token transfers delegates are not transferred. Because of that amount of delegates can be abused.

This may happen:

1. Alice has 10 OpenSwap tokens, which are delegated to Bob.
2. Alice transfers 10 OpenSwap tokens to Max.
3. Max delegates to Bob.

After that steps Bob will have 20 delegates (even totalSupply is 10).

Recommendation: Remove delegation code if it's not going to be used or fix the issue by delegating votes with token transfers.

2. Move delegate after token receive (SushiToken)

In some cases delegate() L75 and delegateBySig() L88 will fail.

1. Alice has 10 OpenSwap tokens, which are delegated to Bob.
2. Alice acquires 1 additional OpenSwap tokens from a transfer.
3. If Alice attempts to re-delegate her 11 OpenSwap tokens to Carol, it will fail due to the SafeMath check on L208; effectively, the function will attempt to undelegate 11 tokens from Bob instead of 10, and revert.

Recommendation: Remove delegation code if it's not going to be used or fix the issue by delegating votes with token transfers.

3. Withdraw amount (TokenLock)

In function payDev() L41 amount of tokens that will be sent to the dev is chosen as minimum of number of OpenSwap tokens that contract has and number of tokens that are allowed to be withdrawn once a week. After that burn mechanism is proceeds. Because of this burn there may be one problem.

For example dev is allowed to withdraw 100 tokens for a week and contract has 50 OpenSwap tokens. Also total supply of "OpenSwap Dev Lock" tokens for example 20 tokens. In this case dev cannot withdraw OpenSwap tokens because contract will try to burn 50 "OpenSwap Dev Lock" tokens and will fail.

4. Duplicate pools (MasterChef)

By the function add() L136 pool can be added with token, that have been already in the other pool. This may brake math of OpenSwap token distribution.

Recommendation: Create a mapping and check if LP pool with the same token was already added.

Low severity issues

1. Old compiler version (All contracts)

Contracts are compiled by 0.6.12 version of solc compiler. It is recommended to use newer version.

2. Gas optimization (TokenLock)

It is a waste of gas to initialize lastPaymentBlocktime variable with zero value.

In function payDev() L41 there is multiple read of global variables OpenSwapToken and weiPerWeek.

Recommendation: Do not initialize variable with zero value.Make an additional local variable to optimise gas usage.

3. Error messages (MasterChef)

In case user sends not valid pid number to functions updatePool() L233, deposit() L257, withdraw() L283 and extWithdraw() L300, he will get unsubstantiated error messages.

Recommendation: Refactor error messages.

4. massUpdatePools problem (MasterChef)

In function massUpdatePools() L225 there is the for loop and number of iterations depends on amount of pools. If the number of pools will be large, then tx may run out-of-gas.

Conclusion

OpenSwap All contracts, SushiToken, SushiMaker, Ownable, TokenLock, MasterChef, UniswapV2ERC20, UniswapV2Router02 contracts were audited. 2 high, 4 medium, 4 low severity issues were found.

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 0xGuard 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 0xGuard 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.

0x Guard