0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

...

OpenSwapV2

October 2021

0xguard.com

hello@0xguard.com

Contents

Introduction

OpenSwapV2 project is a fork of SushiSwap. One of the most significant changes is refactored SushiMaker contract: the main difference is implementation of burning tokens instead of sending to SushiBar.

Name	OpenSwapV2
Audit date	2021-10-18 - 2021-10-20
Language	Solidity
Platform	Binance Smart Chain

Contracts checked

Name	Address
MasterChef	
OpenSwapBridge	
Ownable	
OpenSwapToken	
UniswapV2Pair	

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	not passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	not 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	not passed
Reentrancy	passed

Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	passed
Outdated Compiler Version	passed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. Unrestricted value of OpenSwapPerBlock (MasterChef)

Value of state variable OpenSwapPerBlock is not capped. It's considered as High because the Owner can directly influence on that setting currentBlocktime calling changeBlocktime function (L162).

Recommendation: Consider capping OpenSwapPerBlock by restricting parameter of changeBlocktime function.

Update: The issue was fixed according to recommendation.

2. Sending user's tokens to Collector's address (MasterChef)

In function extWithdraw there is a call safeTokenTransfer(msg.sender, pending); which sends user's OpenSwap tokens to Collector's address since msg.sender is always collector.

Recommendation: Consider sending the tokens to the user instead of collector or add an explanation if it's desired behaviour.

Update: The issue was fixed according to recommendation.

3. pendingOwner is not cancelled (Ownable)

When function transferOwnership (L30) is called with parameter direct equal true, the pendingOwner is not canceled. It makes possible to the pendingOwner reclaim ownership back by calling claimOwnership.

Recommendation: Set pendingOwner to address(0) after setting owner.

Update: The issue was fixed according to recommendation.

4. Minting by non contract account (OpenSwapToken)

Bridge address can be set as a non contract account. In this case it will be able to call bridgeMint unlimited. It's considered as High severity as long as bridgeOpen is set as true. Once one-way closeBridge function is called, closing bridge minting, this issue mitigates.

Update: The issue was fixed by making bridgeAddress settable only once.

5. Wrong fees calculation (UniswapV2Pair)

Implemented _mintFee (L104) function changing fees calculation formula by multiplying result by 3. It seems like misunderstanding UniSwap's fees computation. The original formula calculates equivalent to 1/6th of the growth in sqrt(k) (1/6th of 0.3 is 0.05). So desired 0.15 is 1/2nd of 0.3. That means the desired share of fee (1/2) should be correctly inserted to the UniSwap's formula (see the whitepaper; formulas 6 and 7)

Recommendation: According to the whitepaper presented above, the line 114 can be changed by multiplying by 1(or just remove the operation) instead of 5:

```
uint denominator = rootK.mul(5).add(rootKLast);
```

Update: The issue was fixed according to recommendation.

Medium severity issues

No issues were found

Low severity issues

1. Duplication of code (MasterChef)

Code of functions deposit and extDeposit is almost the same. The difference is only the extDeposit works with a passed user instead of msg.sender, as the deposit does.

Recommendation: Consider to define a new private or internal function, which will be used by both of of functions mentioned above.

2. Unused variable (MasterChef)

State variable devDivisor is never used in calculating.

Update: The issue was fixed.

3. Unreachable code (MasterChef)

In function extWithdraw (L379-L382) there is a checking, which is always false because _amount is always equals 0

```
  uint256 _amount = 0; //HardCoded 0 Amount: Withdraw only Openswap tokens and not LP tokens


if (_amount > 0) {
    user.amount = user.amount.sub(_amount);
    pool.lpToken.safeTransfer(address(msg.sender), _amount);
}
```

Update: The issue was fixed.

4. Variable Default Visibility (OpenSwapBridge)

State variables openswapV1 and openswapV2 are defined with default visibility.

Recommendation: We recommend defining variables' and functions' visibility explicitly to increase readability.

Update: The issue was fixed according to recommendation.

5. closeBridge is not revertible (OpenSwapToken)

Function closeBridge sets bridgeOpen as false and there is no way to change it again.

Conclusion

OpenSwapV2 MasterChef, OpenSwapBridge, Ownable, OpenSwapToken, UniswapV2Pair contracts were audited. 5 high, 5 low severity issues were found.

All high severity and most of the low severity issues were fixed in the update.

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.

x Guard