0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

Degen Haus

January 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for Degen Haus.

Degen Haus project is a DeFi system, which implements Farming and ERC20 token.

Name	Degen Haus
Audit date	2022-01-19 - 2022-01-20
Language	Solidity
Platform	Fantom Network

Contracts checked

Name	Address
MasterChef	https://ftmscan.com/address/0x72A7A3770B4BC999026F3663F1534581E0c59f2a
TripTokenDegenHaus	https://ftmscan.com/address/0xd948efcc99be419ca9bdace89b2bec31edf13adb

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	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	passed
Unprotected SELFDESTRUCT Instruction	passed

Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	not 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Outdated Compiler Version	passed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. Unlimited minting of Degen-tokens (MasterChef)

During the execution of the function `enterStaking`, some amount of degen-tokens is minted. The function `emergencyWithdraw` transfers users liquidity pool tokens but does not burn their degen-tokens, which were minted by `enterStaking`.

The `enterStaking` can be executed to obtain degen-tokens with LP first, then use `emergencyWithdraw` to obtain the staked LP back, and redo this process again.

2. Broken governance mechanism (TripTokenDegenHaus)

The votes in the governance mechanism of the Token can be double-spent (see sushi votes attack).

Recommendation: Move delegates in the `transfer()` function.

Medium severity issues

1. Out of gas issue possibity (MasterChef)

In case of a large number of pools function `massUpdatePools` might be a cause of the "Out of gas error".

2. Reflection tokens support (MasterChef)

The contract does not support reflect liquidity pool tokens since it is not checking the amount of actually transferred tokens.

Low severity issues

1. Duplicate code (MasterChef)

The code of functions `deposit`/`enterStaking` and `withdraw`/`leaveStaking` is duplicated with the only difference that `enterStaking` mints the `degen` tokens and `leaveStaking` burns ones.

Conclusion

Degen Haus MasterChef and TripTokenDegenHaus contracts were audited.

In audited contracts 2 High and 2 Medium Severity issues were found. The MasterChef contract has known vulnerabilities. It also highly depends on the owner's account.

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