HashEx

ThunderSwap

smart contracts audit report

Prepared for:

thunderswap.finance

Authors: HashEx audit team

April 2021

Contents

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and HashEx hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Introduction

HashEx was commissioned by the ThunderSwap team to perform an audit of ThunderSwap smart contracts. The audit was conducted between April 22 and April 27, 2021.

The audited code is located in ThunderSwap github repository. The audit was performed after the commit 2d93d57. The same code is deployed to Binance Smart Chain (BSC) at the addresses of `0xf6135FCb4A0F469DcBb3e6D83520Dc21825A0001`, `0x6FCC6a77ee6F383395c630EEDe1EE928dFF4E331`.

The documentation is available on docs.thunderswap.finance.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts.
- Formally check the logic behind given smart contracts.

Information in this report should be used to understand the risk exposure of smart contracts, and as a guide to improve the security posture of smart contracts by remediating the issues that were identified.
We found out that the code is a fork of PancakeSwap, which is a fork of Uniswap for Binance Smart Chain (BSC). An audit for Uniswap is available [1]. For this reason we focused on the unaudited parts of code, as well as modifications made by ThunderSwap.

Contracts overview

`ThunderSwapERC20.sol`

Implementation of ERC20 token standard with additional permission functionality.
Allows the unlimited token allowance if set to `uint(-1)`.
The same as PancakeERC20.

`ThunderSwapFactory.sol`

The same as PancakeFactory.

`ThunderSwapPair.sol`

The same as PancakePair.

`ThunderSwapRouter.sol`

Same as PancakeRouter (copy of UniswapV2Router02, supports deflationary tokens).

`Libraries & Interfaces`

The same as PancakeSwap's.

Found issues

High severity issues

Not found.

Medium severity issues

ERC20 implementation

ThunderSwapERC20.sol lacks `increaseApproval()` and `decreaseApproval()` functions. These functions mitigate frontrun attacks on the approve function if a user wants to alter previously approved amounts in one transaction (see [2]).

Discrepancy in fees distribution

The fees values in the code and docs are the same as in PancakeSwap. It must be noted that there is a discrepancy in PancakeSwap's documentation regarding the actual fees values. According to documentation [3], swap fee of 0.2% is distributed as follows:

- 0.17% (⅚ of the fee) returns to liquidity pools in the form of a fee reward for liquidity providers,
- 0.03% (⅙ of the fee) is sent to the Treasury.

ThunderSwapPair L109 sets the fee distribution as ¾ to ¼ or 0.15% and 0.05%.

Update: documentation on the site [3] was updated according to the values set in the code (0.15% to the liquidity pools and 0.05% to the ThunderSwap Treasury).

Low severity issues and general recommendations

No low severity issues were introduced with changes in the forked code.

Conclusion

Audited contracts are a copy of the PancakeSwap. No high severity issues were found.
Inconsistency in the fees distribution between documentation and actual values in the code was found. It's caused by the source of the forked code (PancakeSwap).
Update: documentation on the site [3] was updated according to the actual values set in the code.

It must be noted that the audited contracts fork only part of PancakeSwap repositories that implement basic DEX functionality. The farming, timelock, governance and other functionality that present in the forked repositories weren't audited.

References

1. Uniswap audit
2. Attack vector on approve method
3. Swap / Exchange - ThunderSwap

Appendix. Issues' severity classification

We consider an issue critical if it may cause the unlimited losses or breaks the workflow of the contract and could be easily triggered.

High severity issues may lead to the limited losses or break interaction with users or other contracts under very specific conditions.

Medium severity issues do not cause the full loss of functionality, but break the contract logic.

Low severity issues are typically nonoptimal code, unused variables, errors in messages. Usually these issues do not need immediate reactions.