HashEx

# CafeSwap

## smart contract audit report

Prepared for:

cafeswap.finance

Authors: HashEx audit team

March 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 CafeSwap team to perform an audit of CafeSwap smart contracts. The audit was conducted between March 15 and March 19, 2021.

The audited code is located in CafeSwap github repository. The audit was performed for `cafe-swap-core` commit 410d795 and `cafe-swap-periphery` commit f49c867. Code recheck was done at `cafe-swap-core` commit df406b9 and `cafe-swap-periphery` commit e1d04dc.

There was no documentation provided on cafeswap.finance nor on github.com/cafeswap up to 2021-03-19.

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 core and periphery repos are basically mixed copies of SushiSwap (audit available [1], [2]) and PancakeSwap, which is a fork of Uniswap for Binance Smart Chain (BSC). An audit for Uniswap is available [3]. For this reason we focused on the unaudited parts of code, as well as modifications made by CafeSwap.

# Contracts overview

### `core/CafeERC20.sol`

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

### `core/CafeFactory.sol`

The same as SushiSwap's UniswapV2Factory but with an additional public constant `INIT_CODE_PAIR_HASH` (not used in the contract).

### `core/CafePair.sol`

The same as PancakePair but with migration functionality. See the migration issue for more info.

### `periphery/CafeMigrator.sol`

The same as in PancakeSwap.

`periphery/CafeRouter01.sol`

Same as PancakeRouter01 (copy of UniswapV2Router01).

`periphery/CafeRouter02.sol`

Same as PancakeRouter (copy of UniswapV2Router02).

`periphery/CafeLibrary.sol`

Same as UniswapV2Library in SushiSwap or Uniswap. It differs from Pancake's realisation in the amount of fees.

# Found issues

## High severity issues

### Migration is not implemented

CafePair.sol contract contains `mint` function with call to migrator. However, that migrator contract is not implemented in `cafe-swap-core` repo. The existing CafeMigrator.sol contract in `cafe-swap-periphery` doesn't have the `desiredLiquidity` function.

**Update:** CafeSwap team removed migration functionality from CafePair.sol contract in commit df406b9.

### Migration front-run

The contract CafePair.sol has code to enable migration of Liquidity Provide (LP) tokens (the same mechanism as in SushiSwap). This code is intended to migrate LP tokens from another DEX to CafeSwap. In the meantime no implementation is provided in the audited repositories. There is a known frontrun attack that can prevent this migration described in [1]. The attack can be mitigated with a careful deployment process, so the CafeSwap team should be aware of this potential issue.

**Update:** CafeSwap team removed migration functionality from CafePair.sol contract in commit df406b9.

## Medium severity issues

### ERC20 implementation

CafeERC20.sol lacks `increaseApproval` and `decreaseApproval` functions. These functions mitigate frontrun attacks on the approve function if user wants to alter previously approved amount in one transaction (see [4]).

### BEP20 standard

CafeERC20.sol must implement BEP20 token standard as CafeSwap works on BSC. It misses the function `getOwner` required by the standard [5].

**Update**: CafeSwap team provided feedback on that issue that CafeERC20.sol isn't intended to be used on BEP2 chain which is the main use of function `getOwner`.

### Inconsistent parameters of liquidities check in swap function

The contract CafePair.sol has a `require` check at L194 that is intended to ensure that the constant product invariant is not violated (*k* in Uniswap's AMM formula). Also it checks that liquidity providers are paid at least 0.2% fee. At the same time the CafeSwapLibrary.sol contract uses 0.3% as fee value for calculations.

**Update**: CafeSwap team fixed the issue in commit e1d04dc by setting 0.2% fee in CafeSwapLibrary.sol.

## Low severity issues and general recommendations

CafeERC20.sol lacks `require` statements on transfers, e.g. checking amounts to transfer. We recommend adding them as it improves the user's experience by providing descriptive error messages on transaction failures.

# Conclusion

There's no information on protocol fees as none documentation was provided. Therefore we can't comment on the correctness of modifications concerning found inconsistencies made to the Pancakeswap/Sushiswap/Uniswap sources.

The audited repositories are a mix of stripped SushiSwap and PancakeSwap repositories. There is no migration implementation however it is supported in the CafePair.sol. Known migration implementations have high severity issues, so we rated the missing functionality also as high severity to draw attention. Several medium and low severity issues were found.

**Update:** high severity issues and fees calculation issue were fixed in commits e1d04dc and df406b9.

Audit includes recommendations on the code improving and preventing potential attacks.

It must be noted that the audited contracts fork only part of SushiSwap and 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. SushiSwap audit by PeckShield
2. SushiSwap audit by Quantstamp
3. Uniswap audit
4. ERC20 approve issue in simple words
5. BEP20: Tokens on Binance Smart Chain