



BXH DEX(OKExChain) Audit Report



Version 1.0.0

Serial No. 2021080700022024



Presented by Fairyproof

August 7, 2021



FAIRYPROOF



01. Introduction

This document includes the results of the audit performed by the Fairyproof team on the BXH's DEX contracts deployed on OKExChain, at the request of the BXH team.

Audit Start Time:

August 6, 2021

Audit End Time:

August 7, 2021

Audited Source Files' Onchain Address:

`UniswapV2Factory.sol` : 0xff65BC42c10dcC73aC0924B674FD3e30427C7823

`UniswapV2Router02.sol` : 0x56cdDEAa73444498a24E3303333DCAa46fDeD1707

Audited Source Files:

The calculated SHA-256 values for the audited files when the audit was done are as follows:

```
UniswapV2Factory.sol :  
0x31c0de88702d594a7ab8e693b7ec4a394440ee9278ae4f89ed53c43989282cae  
UniswapV2Router02.sol:  
0xafdac5fe7f81af0ff492aceaffc380024adafcfc24d2e8956dc3adcb43644354
```

The goal of this audit is to review BXH's solidity implementation for its DEX application, study potential security vulnerabilities, its general design and architecture, and uncover bugs that could compromise the software in production.

We make observations on specific areas of the code that present concrete problems, as well as general observations that traverse the entire codebase horizontally, which could improve its quality as a whole.

This audit only applies to the specified code, software or any materials supplied by the BXH team for specified versions. Whenever the code, software, materials, settings, environment etc is changed, the comments of this audit will no longer apply.

— Disclaimer

Note that as of the date of publishing, the contents of this report reflect the current understanding of known security patterns and state of the art regarding system security. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk.

The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from offchain sources are not extended by this review either.

Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit is recommended after the issues covered are fixed.

To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

— Methodology

The above files' code was studied in detail in order to acquire a clear impression of how the its specifications were implemented. The codebase was then subject to deep analysis and scrutiny, resulting in a series of observations. The problems and their potential solutions are discussed in this document and, whenever possible, we identify common sources for such problems and comment on them as well.

The Fairyproof auditing process follows a routine series of steps:

1. Code review that includes the following
 - i. Review of the specifications, sources, and instructions provided to Fairyproof to make sure we understand the size, scope, and functionality of the project's source code.
 - ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 - iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Fairyproof describe.
2. Testing and automated analysis that includes the following:
 - i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run the test cases.
 - ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.

3. Best practices review, which is a review of the source code to improve maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

— Structure of the document

This report contains a list of issues and comments on all the above source files. Each issue is assigned a severity level based on the potential impact of the issue and recommendations to fix it, if applicable. For ease of navigation, an index by topic and another by severity are both provided at the beginning of the report.

— Documentation

For this audit, we used the following sources of truth about how the BXH's DEX should work:

Source Code

These were considered the specification, and when discrepancies arose with the actual code behavior, we consulted with the BXH team or reported an issue.

— Comments from Auditor

No vulnerabilities with critical, high, medium or low-severity were found in the above source code.

The comments and discovery only apply to the code deployed and run on the OKExChain blockchain.

02. About Fairyproof

[Fairyproof](#) is a leading technology firm in the blockchain industry, providing consulting and security audits for organizations. Fairyproof has developed industry security standards for designing and deploying blockchain applications.

03. Introduction to BXH's DEX on OKExChain

The DEX application covered by this audit was developed by the BXH team and has been deployed on OKExChain. The application's source code was mainly forked from Uniswap V2's source code. This audit only covered its DEX functions.

04. Major functions of audited code

The audited code was mainly forked from Uniswap V2's source code but has some changes in the handling of collected transaction fees. Here is the major change:

If the switch for transaction fees is turned on, 0.225% of the transaction fee for each transaction will be charged. This charge all belongs to the DEX instead of being all allocated to the liquidity providers.

05. Key points in audit

During the audit we mainly checked:

- whether or not `initCode` worked correctly and did correct calculations,
- whether or not the handling of transaction fees was correct and appropriate,
- whether or not the implementation worked as designed,
- whether or not the addresses of the third-party contracts this application relies on were correct, and
- whether or not there were overlooked issues or risks.

06. Coverage of issues

The issues that the Fairyproof team covered when conducting the audit include but are not limited to the following ones:

- Re-entrancy Attack
- DDos Attack
- Integer Overflow

- Function Visibility
- Logic Vulnerability
- Uninitialized Storage Pointer
- Arithmetic Precision
- Tx.origin
- Shadow Variable
- Design Vulnerability
- Token Issurance
- Asset Security
- Access Control



07. Severity level reference

Every issue in this report was assigned a severity level from the following:

Critical severity issues need to be fixed as soon as possible.

High severity issues will probably bring problems and should be fixed.

Medium severity issues could potentially bring problems and should eventually be fixed.

Low severity issues are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

08. List of issues by severity

A. Critical



- N/A

B. High

- N/A

C. Medium

- N/A

D. Low

- N/A

09. List of issues by source file

- N/A

10. Issue descriptions

- N/A

11. Recommendations to enhance the overall security

We list some recommendations in this section. They are not mandatory but will enhance the overall security of the system if they are adopted.

- Refine Code Implementation

In the `UniswapV2Factory.sol` contract file, the `pairCodeHash` function uses the following directive:

```
return keccak256(type(UniswapV2Pair).creationCode);
```

consider using the following one instead:

```
return keccak256(abi.encodePacked(type(UniswapV2Pair).creationCode));
```