



AliumSwap AMM
SMART CONTRACT AUDIT

13.04.2021

Made in Germany by Chainsulting.de



Table of contents

1. Disclaimer	3
2. About the Project and Company	4
2.1 Project Overview	5
3. Vulnerability & Risk Level.....	6
4. Auditing Strategy and Techniques Applied	7
4.1 Methodology.....	7
4.2 Used Code from other Frameworks/Smart Contracts	8
4.3 Tested Contract Files.....	10
4.4 Metrics / CallGraph	11
4.6 Metrics / Capabilities.....	13
4.7 Metrics / Source Unites in Scope.....	14
5. Scope of Work.....	18
5.1 Manual and Automated Vulnerability Test	19
5.2. SWC Attacks.....	20
5.3. Associated audits with the forked codebase.....	23
6. Executive Summary	24
7. Deployed Smart Contract.....	24



1. Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of AliumSwap. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

Major Versions / Date	Description
0.1 (30.03.2021)	Layout
0.5 (06.04.2021)	Test Deployment and compare with PancakeSwap
0.6 (07.04.2021)	Testing SWC Checks
0.8 (08.04.2021)	Automated Security Testing Manual Security Testing
0.9 (09.04.2021)	Summary and Recommendation
1.0 (13.04.2021)	Final document

2. About the Project and Company

Company address: NA (ANON)

Website: <http://alium.finance>

Twitter: <http://twitter.com/alium.finance>

Medium: <https://aliumswap.medium.com>

Telegram (ENG): https://t.me/aliumswap_official

Telegram (RU): https://t.me/aliumswap_ru

Telegram (JP): https://t.me/aliumswap_jp

Reddit: https://www.reddit.com/user/AliumSwap_Official

Discord: <https://discord.gg/BU6m6zgpvZ>

LinkedIn: <https://www.linkedin.com/company/75861509>

Documentation: <https://aliumswap.gitbook.io/alium-finance/>

2.1 Project Overview

AliumSwap is a decentralized AMM Exchange with multi-blockchain option and NFTs. Aliumswap claims to be the first AMM DEX with multi-chain support, which starts on the Binance Smart Chain blockchain (BSC). BSC is very similar in its user properties to Ethereum, but the fees are currently hundreds of times lower.

AMM DEX Alium (AliumSwap) allows users to reduce the costs of commissions, they have improved and added some features that are not available on other exchanges on Binance Smart Chain. For example, they allow the user the ability to create any trading pair on their own, without unnecessary coordination with the exchange support.

3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 – 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

4.1 Methodology

The 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 Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
 - 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 Chainsulting 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 those test cases.
 - ii. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.



4.2 Used Code from other Frameworks/Smart Contracts (direct imports / similar codebase)

Dependency / Import Path	Source
IAliumERC20.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/interfaces/IPancakeERC20.sol
IAliumFactory.sol	https://github.com/Uniswap/uniswap-v2-core/blob/master/contracts/interfaces/IUniswapV2Factory.sol https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/interfaces/IPancakeFactory.sol
IAliumPair.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/interfaces/IPancakePair.sol
SafeMath.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/libraries/SafeMath.sol
AliumERC20.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/PancakeERC20.sol
Math.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/libraries/Math.sol
UQ112x112.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/libraries/UQ112x112.sol
IAliumCallee.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/interfaces/IPancakeCallee.sol
AliumPair.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/PancakePair.sol
AliumFactory.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/PancakeFactory.sol
AliumMigrator.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/PancakeMigrator.sol
IAliumMigrator.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IPancakeMigrator.sol

Dependency / Import Path	Source
IAliumRouter01.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IPancakeRouter01.sol
IAliumRouter02.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IPancakeRouter02.sol
IWETH.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IWETH.sol
AliumRouter.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/PancakeRouter.sol
AliumRouter01.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/PancakeRouter01.sol
AliumOracleLibrary.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/libraries/PancakeOracleLibrary.sol https://github.com/Uniswap/uniswap-lib/blob/master/contracts/libraries/FixedPoint.sol
AliumLibrary.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/libraries/PancakeLibrary.sol

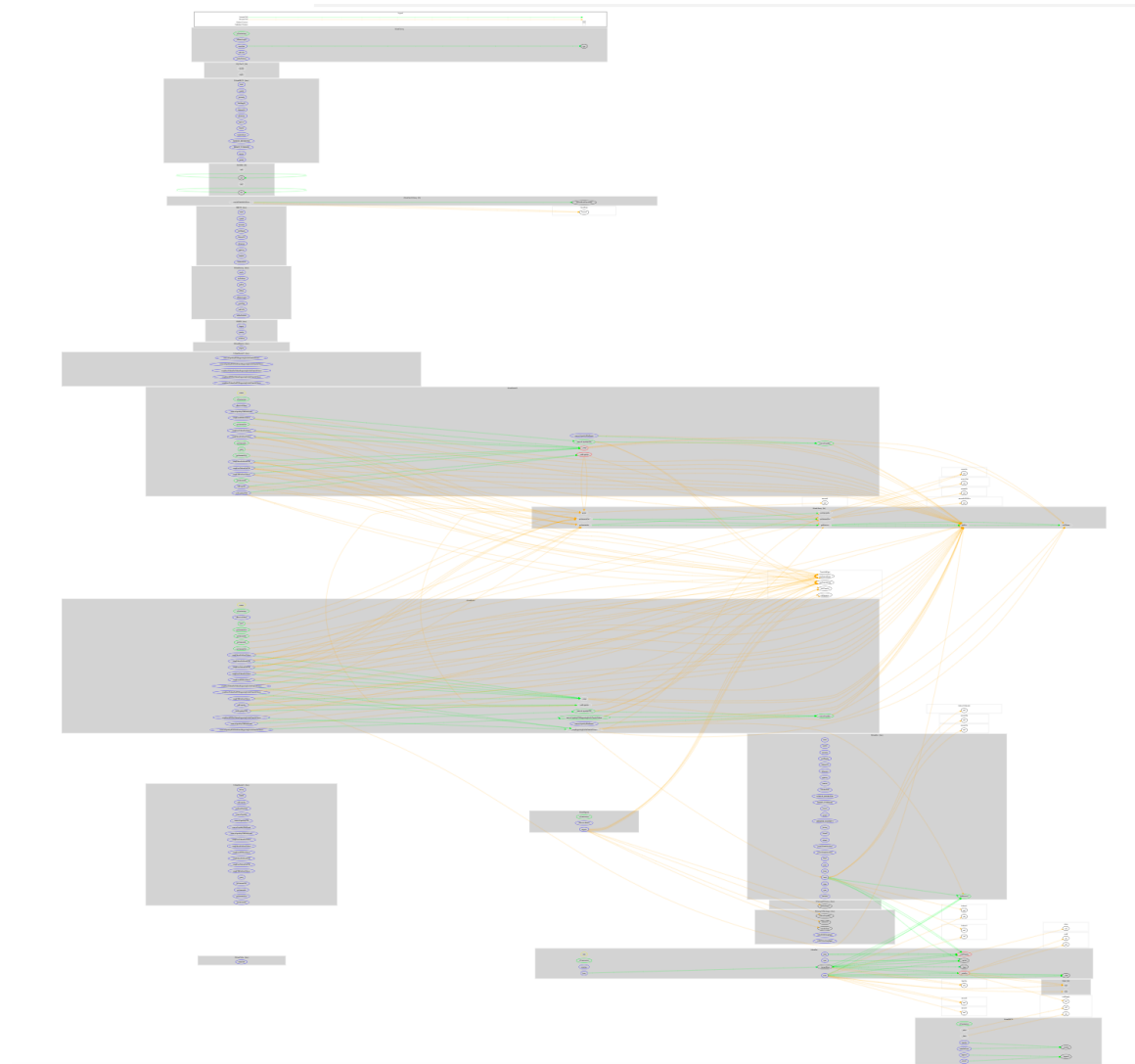
4.3 Tested Contract Files

The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

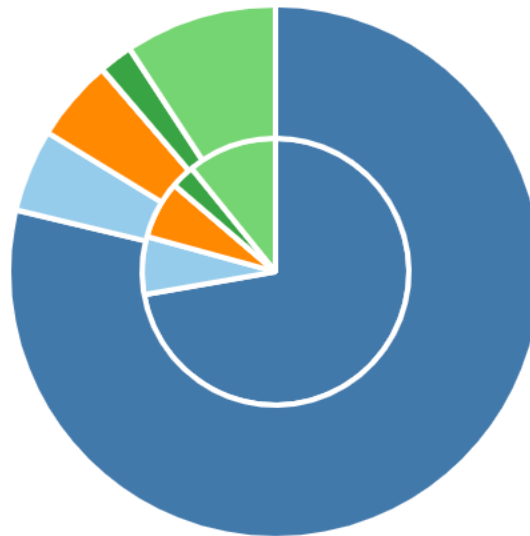
File	Fingerprint (MD5)
alium-swap-core/contracts/AliumFactory.sol	04bdfd9d13bf965d4eb934641c5dd214
alium-swap-core/contracts/AliumERC20.sol	a36f7f0c751811f3fea983c2e7205ed0
alium-swap-core/contracts/AliumPair.sol	e9af9ef35a89e07e1a1a2b82ed82feac
alium-swap-core/contracts/interfaces/IAliumCallee.sol	b73f14a0e4ea1640cd00a2c1878fda5e
alium-swap-core/contracts/interfaces/IAliumERC20.sol	6a4adff918f6c0e698810bcb9f7d4151
alium-swap-core/contracts/interfaces/IAliumFactory.sol	50c7a014b93c90abb427f880df731cb2
alium-swap-core/contracts/interfaces/IAliumPair.sol	28a1795879dd36c328069d253279687f
alium-swap-core/contracts/interfaces/IERC20.sol	3d2b948a9266ee66b6e739de97c7e6e9
alium-swap-core/contracts/libraries/Math.sol	ae17cd43b5fec791f48a7c42cf084636
alium-swap-core/contracts/libraries/SafeMath.sol	109a009c7606bc3d38666494bd593c19
alium-swap-core/contracts/libraries/UQ112x112.sol	413dce52dbc8344bb43fc67a2278dd99
alium-swap-periphery-master/contracts/AliumMigrator.sol	7f4f984b82de2e45bd595aa60293883a
alium-swap-periphery-master/contracts/AliumRouter.sol	108158016840895a3160bc172fe64d1c
alium-swap-periphery-master/contracts/AliumRouter01.sol	72fbc48634f3627cde8198167997c264
alium-swap-periphery-master/contracts/libraries/AliumLibrary.sol	6e7d21b609e5df4063a51a52ab4c4501
alium-swap-periphery-master/contracts/libraries/AliumOracleLibrary.sol	ec447d76c709fe9d98a1afdf6d81149c
alium-swap-periphery-master/contracts/interfaces/IAliumRouter01.sol	4f81da7aa121758ad06e095b74941284
alium-swap-periphery-master/contracts/interfaces/IAliumRouter02.sol	d838e97b42e8303782972eba1047be70
alium-swap-periphery-master/contracts/interfaces/IAliumMigrator.sol	ec556996c2f994f73326acda0b290c16

4.4 Metrics / CallGraph











Full Version: https://chainsulting.de/wp-content/uploads/2021/04/aliumswap_solidity-metrics.html





4.5 Metrics / Source Lines




4.6 Metrics / Capabilities











Solidity Versions observed		 Experimental Features	 Can Receive Funds	 Uses Assembly	 Has Destroyable Contracts
<code>>=0.5.0</code> <code>=0.6.6</code>			yes	yes (2 asm blocks)	
 Transfers ETH	 Low-Level Calls	 DelegateCall	 Uses Hash Functions	 ECRrecover	 New/Create/Create2
yes			yes	yes	yes → <code>AssemblyCall::Name::create2</code>














 Public	 Payable			
201	16			
External	Internal	Private	Pure	View
179	141	7	43	66











StateVariables


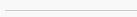
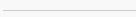




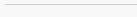






Total	 Public
33	25

4.7 Metrics / Source Unites in Scope

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	alium-swap-periphery-master/contracts/interfaces/IAliumallee.sol	_____	1	5	4	3	_____	3	_____
	alium-swap-periphery-master/contracts/interfaces/IAliumRouter01.sol	_____	1	95	4	3	_____	48	
	alium-swap-periphery-master/contracts/interfaces/IAliumPair.sol	_____	1	52	7	5	_____	55	_____
	alium-swap-periphery-master/contracts/interfaces/IAliumRouter02.sol	_____	1	44	6	4	_____	16	
	alium-swap-periphery-master/contracts/interfaces/IAliumMigrator.sol	_____	1	5	4	3	_____	3	_____
	alium-swap-periphery-master/contracts/interfaces/IWETH.sol	_____	1	7	4	3	_____	10	
	alium-swap-periphery-master/contracts/interfaces/IAliumFactory.sol	_____	1	17	6	4	_____	17	_____

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	alium-swap-periphery-master/contracts/interfaces/IERC20.sol		1	17	7	5		19	
	alium-swap-periphery-master/contracts/interfaces/V1/IUniswapV1Exchange.sol		1	9	4	3		14	
	alium-swap-periphery-master/contracts/interfaces/V1/IUniswapV1Factory.sol		1	5	4	3		3	
	alium-swap-periphery-master/contracts/AliumRouter01.sol	1		280	191	172	9	210	
	alium-swap-periphery-master/contracts/libraries/AliumOracleLibrary.sol	1		35	33	20	8	14	
	alium-swap-periphery-master/contracts/libraries/SafeMath.sol	1		17	17	12	1	4	
	alium-swap-periphery-master/contracts/libraries/AliumLibrary.sol	1		83	83	64	9	72	
	alium-swap-periphery-master/contracts/AliumRouter.sol	1		447	287	259	14	310	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	alium-swap-periphery-master/contracts/AliumMigrator.sol	1		49	46	37	4	33	
	alium-swap-core-master/contracts/interfaces/IAliumallee.sol		1	5	4	3		3	
	alium-swap-core-master/contracts/interfaces/IAliumPair.sol		1	52	7	5		55	
	alium-swap-core-master/contracts/interfaces/IAliumERC20.sol		1	23	7	5		27	
	alium-swap-core-master/contracts/interfaces/IAliumFactory.sol		1	17	6	4		17	
	alium-swap-core-master/contracts/interfaces/IERC20.sol		1	17	7	5		19	
	alium-swap-core-master/contracts/AliumPair.sol	1		222	222	186	36	199	
	alium-swap-core-master/contracts/libraries/Math.sol	1		23	23	18	2	5	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	alium-swap-core-master/contracts/libraries/UQ112x112.sol	1		20	20	10	6	4	
	alium-swap-core-master/contracts/libraries/SafeMath.sol	1		34	34	23	3	7	
	alium-swap-core-master/contracts/AliumFactory.sol	1		51	51	41	2	53	
	alium-swap-core-master/contracts/AliumERC20.sol	1		94	94	79	1	61	
	Totals	12	15	1725	1182	979	95	1281	

Legend: []

- **Lines:** total lines of the source unit
- **nLines:** normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC:** normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines:** lines containing single or block comments
- **Complexity Score:** a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ..)

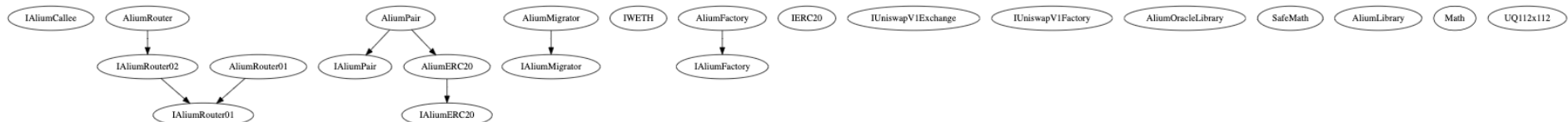
5. Scope of Work

The AliumSwap Team provided us with the files that needs to be tested. The scope of the audit are the AliumSwap AMM Protocol contracts.

Following contracts with the direct imports has been tested:

- AliumFactory.sol
- AliumERC20.sol
- AliumRouter.sol
- AliumMigrator.sol

The main goal of this audit was to verify the overall smart contract security and code quality.



5.1 Manual and Automated Vulnerability Test

CRITICAL ISSUES

During the audit, Chainsulting's experts found **no Critical issues** in the code of the smart contract.

HIGH ISSUES

During the audit, Chainsulting's experts found **no High issues** in the code of the smart contract.

MEDIUM ISSUES

During the audit, Chainsulting's experts found **no Medium issues** in the code of the smart contract.

LOW ISSUES

During the audit, Chainsulting's experts found **no Low issues** in the code of the smart contract.

INFORMATIONAL ISSUES





During the audit, Chainsulting's experts found **no Informational issues** in the code of the smart contract.

5.2. SWC Attacks

ID	Title	Relationships	Test Result
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	
SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	

ID	Title	Relationships	Test Result
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	✓
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	✓
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	✓
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	✓
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	✓
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	✓
SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✓
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	✓
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	✓

ID	Title	Relationships	Test Result
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	✓
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✓
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	✓
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	✓
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	✓
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	✓
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	✓
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	✓
SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	✓
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	✓

ID	Title	Relationships	Test Result
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	

5.3. Associated audits with the forked codebase

PancakeSwap (Factory / Router / Pair):

<https://www.certik.org/projects/pancakeswap> (13.10.2020)

6. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase. As most of the used code was forked from widely used (Uniswap/PancakeSwap) and audited codebases, the attack surface is pretty small. All modifications from the codebase that diff. from the original source, have been reviewed and approved.

The main goal of the audit was to verify the security of the smart contract and the functions. During the audit, no issues were found, after the manual and automated security testing.

7. Deployed Smart Contract

VERIFIED

Smart Contract is deployed here:

Factory

<https://bscscan.com/address/0xbEAC7e750728e865A3cb39D5ED6E3A3044ae4B98#code>

Router

<https://bscscan.com/address/0xB0e28C53B7C84741085EFE2e16CFF1d04149848f#code>