



MakiSwap AMM
SMART CONTRACT AUDIT

05.04.2021

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.....	11
4.4 Metrics / CallGraph	12
4.6 Metrics / Capabilities.....	14
4.7 Metrics / Source Unites in Scope.....	15
5. Scope of Work.....	18
5.1 Manual and Automated Vulnerability Test	19
5.1.1 Linter parse error (())	19
5.1.2 Wrong import of OpenZeppelin library.....	20
5.1.3 Multiple code lines with the same functions	21
5.1.4 Fractions from BSC / Pancakeswap Fork	22
5.1.5 Interfaces are inside the main code	23
5.1.6 Two files have almost the same code	23
5.1.7 Use of different pragma versions	24
5.2. SWC Attacks	24
5.3. Associated Audits with the copied codebase	28



6. Executive Summary	29
7. Deployed Smart Contract.....	29

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 MakiSwap (The Unilayer Project Ltd). 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 (23.03.2021)	Layout
0.5 (23.03.2021)	Verify Claims and Test Deployment
0.6 (24.03.2021)	Testing SWC Checks
0.8 (24.03.2021)	Automated Security Testing Manual Security Testing
0.9 (25.03.2021)	Summary and Recommendation
1.0 (26.03.2021)	Reported issues and delivery of new codebase
1.1 (26.03.2021)	Re-check
1.5 (26.03.2021)	Final document
1.6 (30.03.2021)	Re-check new contract repo
2.0 (05.04.2021)	Re-check of contracts and recreation of audit report

2. About the Project and Company

Company address:

The Unilayer Project Ltd
44 Church St
St John's
Antigua & Barbuda

Website: <https://www.makiswap.com>

Twitter: <https://twitter.com/makiswap>

Telegram: <https://t.me/MakiSwap>

2.1 Project Overview

MakiSwap is an AMM and Yield Farming platform built on Huobi Eco Chain (HECO) and part of the growing UniLayer Ecosystem. MakiSwap was developed by the Unilayer team as a foundation, laying down this foundation is essential for Blockchain Interoperability between Ethereum and HECO, MakiSwap will make things easier to seamlessly integrate all of the UniLayer DeFi tools on HECO!

What are some of the benefits for Liquidity Providers on MakiSwap?

It's easy to become to a Liquidity Provider on MakiSwap, simply add liquidity into a pool and earn rewards in the form of MAKI tokens. However, unlike Uniswap or other protocols those MAKI tokens will also allow you to earn a portion of the protocol's fee, accumulated in MAKI, even if you decide that you no longer want to participate in the liquidity provision. As an early adopter to help provider liquidity, you become a significant stakeholder of the protocol.

The rewards you will receive from staking will be proportional to the amount of LP tokens you have staked against the total amount of LP tokens staked.

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
@openzeppelin/contracts/access/Ownable.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/access/Ownable.sol
@openzeppelin/contracts/math/SafeMath.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/math/SafeMath.sol
@openzeppelin/contracts/GSN/Context.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/GSN/Context.sol
@openzeppelin/contracts/token/ERC20/IERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/token/ERC20/IERC20.sol
@openzeppelin/contracts/utils/Address.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/utils/Address.sol
@openzeppelin/contracts/token/ERC20/ERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/token/ERC20/ERC20.sol
@openzeppelin/contracts/token/ERC20/SafeERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.2.0/contracts/token/ERC20/SafeERC20.sol
Multicall.sol	https://github.com/makerdao/multicall/blob/master/src/Multicall.sol
SoyBar.sol (Governance)	https://github.com/yam-finance/yam-protocol/blob/master/contracts/token/YAMGovernance.sol
SoyBar.sol (Governance)	https://github.com/yam-finance/yam-protocol/blob/master/contracts/token/YAMGovernanceStorage.sol
MasterChef.sol	https://github.com/sushiswap/sushiswap/blob/master/contracts/MasterChef.sol

Dependency / Import Path	Source
IMakiswapFactory.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
IMakiswapPair.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
MakiswapHRC20.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
IMakiswapCallee.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/interfaces/IPancakeCallee.sol
MakiswapPair.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/PancakePair.sol
MakiswapFactory.sol	https://github.com/pancakeswap/pancake-swap-core/blob/master/contracts/PancakeFactory.sol
TransferHelper.sol	@uniswap/lib/contracts/libraries/TransferHelper.sol

Dependency / Import Path	Source
IMakiswapRouter01.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IPancakeRouter01.sol
IMakiswapRouter02.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IPancakeRouter02.sol
IWHT.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/interfaces/IWETH.sol
MakiswapRouter.sol	https://github.com/pancakeswap/pancake-swap-periphery/blob/master/contracts/PancakeRouter.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)
./contracts/exchange/MakiswapFactory.sol	2756d1605d1fa392c23e945fbec2e6b2
./contracts/exchange/MakiswapHRC20.sol	c4f686a3da8dba5e57a7ceb48bf3a13e
./contracts/exchange/MakiswapPair.sol	41ceec569a7175f0f3737491431239f1
./contracts/exchange/MakiswapRouter.sol	f211a5ba1a2475a61d93122047f62b8b
./contracts/exchange/interfaces/IHRC20.sol	57e8ad4b5c1f661619ed6241984f8a2b
./contracts/exchange/interfaces/IMakiswapCallee.sol	230c806c50c731ba518d7628759ba126
./contracts/exchange/interfaces/IMakiswapFactory.sol	27e5f2aad03b151c0d8c948513ebb267
./contracts/exchange/interfaces/IMakiswapHRC20.sol	ed3a2591001cf0aa8d3290c616ea509c
./contracts/exchange/interfaces/IMakiswapPair.sol	036fbd8dee3d8a058767795ed81c908f
./contracts/exchange/interfaces/IMakiswapRouter01.sol	61ecc931d2659c48cee34a0b9eedc649
./contracts/exchange/interfaces/IMakiswapRouter02.sol	cf0c06a2ebf17d6c49324bebc77c733e
./contracts/exchange/interfaces/IWHT.sol	0ac3df74d887609b4d33e8ad3bfb2d0b
./contracts/exchange/libraries/MakiswapLibrary.sol	66c419e3c7472f5aaadb6d154b873f78
./contracts/exchange/libraries/Math.sol	d28ea57c6a55382feca441233abab1f6
./contracts/exchange/libraries/SafeMath.sol	a921a39062f92d470483ebe52f31c328
./contracts/exchange/libraries/UQ112x112.sol	70e94f6712a33500d3ab4a5fbc7de1cd
./contracts/exchange/utils/TransferHelper.sol	170c61c23d375378f988bd1da7901fce
./contracts/farm/MakiToken.sol	3a20cfe59b8ac1bf4319c9971fd7983c
./contracts/farm/MasterChef.sol	de208d2138637048a0763efdacab5dae
./contracts/farm/SousChef.sol	88c73be2cc50fad080b40afbbc4e11bc
./contracts/farm/SoyBar.sol	af3881bfd16db8fb8d0fc590cc71bd3c
./contracts/farm/libraries/Multicall.sol	5f0c1a25ff8276331a264e6d35efd7b9
./contracts/farm/token/HRC20.sol	f21fc5806aec375e6302d02d7816a546
./contracts/farm/token/SafeHRC20.sol	47a1165ae31209d223ecfb8f4014f82c
./contracts/farm/token/WHT.sol	1146f86fe4d53ff85496d1cb99dd3331

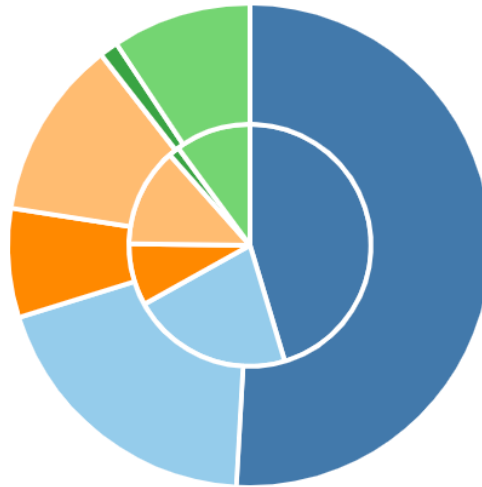
4.4 Metrics / CallGraph

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















4.5 Metrics / Source Lines

source comment single block mixed
empty todo blockEmpty



































4.6 Metrics / Capabilities













Solidity Versions observed		 Experimental Features	 Can Receive Funds	 Uses Assembly	 Has Destroyable Contracts
<pre>>=0.5.0 >=0.5.16 =0.5.16 >=0.6.6 >=0.5.6 >=0.5.0 <0.8.0 ^0.6.12 0.6.12 >=0.4.0</pre>		ABIEncoderV2	yes	yes (6 asm blocks)	
 Transfers ETH	 Low-Level Calls	 DelegateCall	 Uses Hash Functions	 ECRRecover	 New/Create/Create2
yes			yes	yes	yes → AssemblyCall:Name:create2

 Public	 Payable			
208	12			
External	Internal	Private	Pure	View
143	220	7	51	73

4.7 Metrics / Source Unites in Scope

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/exchange/interfaces/IHRC20.sol	_____	1	21	11	5	2	19	_____
	contracts/exchange/interfaces/IMakiswapCallee.sol	_____	1	11	6	3	1	3	_____
	contracts/exchange/interfaces/IMakiswapPair.sol	_____	1	54	9	5	1	55	_____
	contracts/exchange/interfaces/IMakiswapFactory.sol	_____	1	20	9	4	2	17	_____
	contracts/exchange/interfaces/IMakiswapHRC20.sol	_____	1	25	9	5	1	27	_____
	contracts/exchange/interfaces/IMakiswapRouter01.sol	_____	1	99	8	3	2	48	
	contracts/exchange/interfaces/IWHT.sol	_____	1	11	8	3	2	10	
	contracts/exchange/interfaces/IMakiswapRouter02.sol	_____	1	47	9	4	2	16	
	contracts/exchange/MakiswapRouter.sol	1	_____	456	296	257	23	310	 

Typ e	File	Logic Contracts	Interfaces	Lin es	nLin es	nSL OC	Comm ent Lines	Compl ex. Score	Capabilitie s
	contracts/exchange/utils/TransferHelper.sol	1	_____	42	29	19	5	26	_____
	contracts/exchange/libraries/Math.sol	1	_____	27	27	18	4	5	_____
	contracts/exchange/libraries/UQ112x112.sol	1	_____	23	23	10	8	4	_____
	contracts/exchange/libraries/SafeMath.sol	1	_____	191	179	54	108	14	_____
	contracts/exchange/libraries/MakiswapLibrary.sol	1	_____	90	90	64	14	72	
	contracts/exchange/MakiswapFactory.sol	1	_____	68	68	49	6	54	
	contracts/exchange/MakiswapHRC20.sol	1	_____	96	96	77	6	59	
	contracts/exchange/MakiswapPair.sol	1	1	223	220	174	42	194	
	contracts/farm/interfaces/IHRC20.sol	_____	1	21	11	5	2	19	_____
	contracts/farm/MasterChef.sol	1	1	365	356	246	75	212	_____
	contracts/farm/SoyBar.sol	1	_____	270	240	148	54	96	
	contracts/farm/MakiToken.sol	1	_____	243	213	129	52	79	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/farm/libraries/Multicall.sol	1	_____	47	47	38	6	37	
	contracts/farm/libraries/Address.sol	1	_____	163	130	57	88	37	
	contracts/farm/libraries/SafeMath.sol	1	_____	189	177	54	107	14	_____
	contracts/farm/SousChef.sol	1	_____	171	171	112	36	80	_____
	contracts/farm/token/HRC20.sol	1	_____	319	307	108	169	91	_____
	contracts/farm/token/WHT.sol	1	_____	64	61	47	1	39	
	contracts/farm/token/SafeHRC20.sol	1	_____	99	77	35	32	25	_____
	Totals	19	11	3455	2887	1733	851	1662	

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, ..)

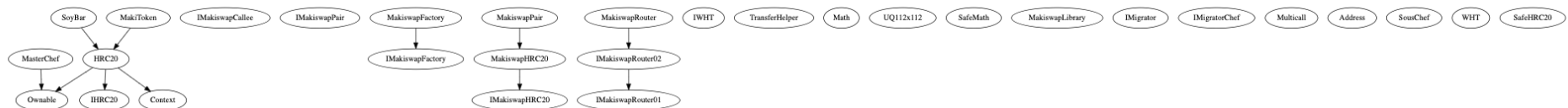
5. Scope of Work

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

Following contracts with the direct imports has been tested:

- MakiwapFactory.sol
- MakiwapHRC20.sol
- MakiwapRouter.sol
- MasterChef.sol
- SousChef.sol
- SoyBar.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

5.1.1 Linter parse error (())

Severity: MEDIUM

Status: **FIXED**

File(s) affected: MasterChef.sol

Attack / Description	Code Snippet	Result/Recommendation
The contract is using names, which are used within the Binance Smart Chain ecosystem and smart contract structure. Most probably fractions from the fork process.	Line 1658 - 1659 maki.mint(devaddr, makiReward.mul(15).div(130); maki.mint(burnaddr,makiReward.mul(15).div(130);	We recommended closing the brackets. maki.mint(devaddr, makiReward.mul(15).div(130)); maki.mint(burnaddr,makiReward.mul(15).div(130));

LOW ISSUES

5.1.2 Wrong import of OpenZeppelin library

Severity: LOW

Status: **FIXED**

<https://github.com/MakiSwap-Protocol/makiswap-protocol>

File(s) affected: All

Attack / Description	Code Snippet	Result/Recommendation
In the current implementation, OpenZeppelin files are added directly into the code. This violates OpenZeppelin's MIT license, which requires the license and copyright notice to be included if its code is used. Moreover, updating code manually is error-prone.	NA	We highly recommend using npm (import "@openzeppelin/contracts/..") in order to guarantee that original OpenZeppelin contracts are used with no modifications. This also allows for any bug-fixes to be easily integrated into the codebase.

5.1.3 Multiple code lines with the same functions

Severity: LOW

Status: **FIXED**

<https://github.com/MakiSwap-Protocol/makiswap-protocol>

File(s) affected: MakiBar.sol, MakiToken.sol, MasterChef.sol

Attack / Description	Code Snippet	Result/Recommendation
It is not necessary to have two times or more the same functions inside different files.	<p>Line: 936 – 1193 (MakiBar.sol)</p> <pre>contract MakiToken is BEP20("MakiSwap Token", "MAKI") {</pre> <p>Line: 1225 – 1428 (MakiBar.sol)</p> <pre>contract MakiBar is BEP20("MakiBar Token", "xMAKI") {</pre> <p>// Copied and modified from YAM code: // https://github.com/yam-finance/yam-protocol/blob/master/contracts/token/YAMGovernanceStorage.sol // https://github.com/yam-finance/yam-protocol/blob/master/contracts/token/YAMGovernance.sol // Which is copied and modified from COMPOUND: // https://github.com/compound-finance/compound-protocol/blob/master/contracts/Governance/Comp.sol</p> <p>Line: 929 - 1193 (MakiToken.sol)</p> <p>Line: 1137 - 1405 (MasterChef.sol)</p>	We highly recommend creating libraries if a function is used multiple times. For a better code quality and smaller size of the code.

INFORMATIONAL ISSUES

5.1.4 Fractions from BSC / Pancakeswap Fork

Severity: INFORMATIONAL

Status: **FIXED**

<https://github.com/MakiSwap-Protocol/makiswap-protocol/commit/a816f17ad3f559468997e1a6e9026bd2f6e11b9e>

File(s) affected: MakiBar.sol, MakiToken.sol, MasterChef.sol,

Attack / Description	Code Snippet	Result/Recommendation
The contract is using names, which are used within the Binance Smart Chain ecosystem and smart contract structure. Most probably fractions from the fork process.	Line 109 <code>interface IBEP20</code> Line 604 <code>contract BEP20 is Context, IBEP20, Ownable {</code> Line 796 <code>"BEP20: decreased allowance below zero"</code> Line 929 <code>contract MakiToken is BEP20("MakiSwap Token", "MAKI")</code> And lot more code lines / comments.	For a better code quality, it is recommended to use different names ex. IBEP20 = IHRC20. Search and replace "BEP" with "HRC".

5.1.5 Interfaces are inside the main code

Severity: INFORMATIONAL

Status: **FIXED**

<https://github.com/MakiSwap-Protocol/makiswap-protocol>

File(s) affected: swapV2Factory.sol

Attack / Description	Code Snippet	Result/Recommendation
Interfaces are always better to separate into different files.	Line: 3 - 17 <code>interface IMakiFactory</code> Line: 19 - 68 <code>interface IMakiPair</code> For more interfaces see 4.2	It is best practice to have interfaces not in the main codebase and better under ./interfaces/..

5.1.6 Two files have almost the same code

Severity: INFORMATIONAL

Status: **FIXED**

<https://github.com/MakiSwap-Protocol/makiswap-protocol>

File(s) affected: swapV2Router02.sol / WHT.sol

Attack / Description	Code Snippet	Result/Recommendation
Same content in two files	Line: 263 diff: hex 'd0d4c4cd0848c93cb4fd1f498d7013ee6bfb25783ea21593d5834f5d250ece66' // init code hash	Remove that file from repo, as duplicated code

5.1.7 Use of different pragma versions

Severity: INFORMATIONAL

Status: ADDRESSED

File(s) affected: All










Attack / Description	Code Snippet	Result/Recommendation
Source files requires different compiler version.	<pre>>=0.5.0 >=0.5.16 =0.5.16 >=0.6.6 >=0.5.6 >=0.5.0 <0.8.0 ^0.6.12 0.6.12 >=0.4.0</pre>	Consider deploying with 0.6.11 and use a constant version at all files.

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	✓

ID	Title	Relationships	Test Result
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	✓
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	✓

ID	Title	Relationships	Test Result
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')	✓
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	✓

ID	Title	Relationships	Test Result
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	
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 copied codebase

PancakeSwap (Factory / Router / Pair):

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

SushiSwap (MasterChef):

https://github.com/peckshield/publications/blob/master/audit_reports/PeckShield-Audit-Report-SushiSwap-v1.0.pdf (09.03.2020)

YAM Finance (Governance):

<https://github.com/yam-finance/yamV3/blob/master/PeckShield-Audit-Report-YAMv3-v1.0.pdf> (10.03.2020)

6. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase. We recommend separating the libraries and interfaces out of the flattened codebase, for a better code quality and smaller size. As most of the used code was copied from widely used (SushiSwap, YAM Finance, Uniswap/PancakeSwap, OpenZeppelin) 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 critical issues were found, after the manual and automated security testing. Only issues that were found, are regards the code quality.

Update 30.03.2021: libraries have been separated from the codebase and code quality has improved as well. All addressed issues has been fixed.

Update 05.04.2021: latest modification (<https://github.com/MakiSwap-Protocol/makiswap-protocol/commit/61659f8046092a1836e1d988ae2d9a84fe71372f>) has been checked and audit got recreated.

7. Deployed Smart Contract

PENDING

Smart Contract is deployed here:
0xFF

Open Source Code:
<https://github.com/MakiSwap-Protocol/maki-farm>
<https://github.com/MakiSwap-Protocol/maki-swap>

