



1Inch Mooniswap V2 & 1INCH Token

SMART CONTRACT AUDIT

25.12.2020

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	9
4.4 Metrics / CallGraph.....	10
4.5 Metrics / Source Lines	14
4.6 Metrics / Capabilities	15
4.7 Metrics / Source Unites in Scope	16
5. Scope of Work.....	18
5.1 Manual and Automated Vulnerability Test.....	19
5.1.1 Cannot pause / shutdown Mooniswap Factory	19
5.1.2 Potential violation of Checks-Effects-Interaction pattern	20
5.1.3 Potential front running attack or losing of allowance	21
5.1.4 Mismatch of argument name in _PERMIT_TYPEHASH and permit function	22
5.1.6 Missing natspec documentation.....	24
5.1.7 Fix Spelling and Grammatical Errors	25
5.1.8 A floating pragma is set.....	26
5.1.9 "dist:factory": "truffle-flattener Mooni*" (rename)	27



6. Executive Summary.....	28
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 1Inch Exchange. 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 (25.11.2020)	Layout
0.5 (02.12.2020)	Automated Security Testing Manual Security Testing
0.8 (02.12.2020)	Testing SWC Checks
1.0 (03.12.2020)	Summary and Recommendation
2.0 (03.12.2020)	Final document
2.1 (10.12.2020)	Recheck of codebase
2.5 (15.12.2020)	Verify findings
2.9 (23.12.2020)	Acknowledge of 3th party findings
3.0 (25.12.2020)	Final document

2. About the Project and Company

Company address:

1Inch Limited
Quijano Chambers, P.O. Box 3159, Road Town
Tortola, British Virgin Islands

Sergej Kunz Co-Founder & Chief Executive Officer
Anton Bukov Co-Founder & Chief Technology Officer

Website: <https://mooniswap.exchange/>

GitHub: <https://github.com/CryptoManiacsZone>

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

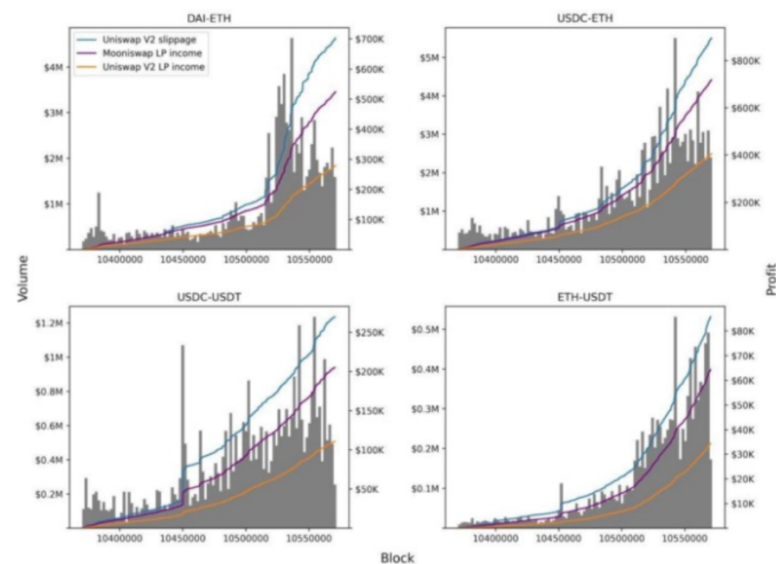
Discord: <https://discord.gg/FZADkCZ>

Youtube: <https://www.youtube.com/channel/UCk0nvK4bHpteQXZKv7lkq5w>

Medium: <https://medium.com/@1inch.exchange>

2.1 Project Overview

Mooniswap is an implementation of an “Automated Market Maker” (AMM), allowing exchange between two assets using an automated pricing mechanism based on balance ratios. While similar in approach to simple AMMs, such as Uniswap or Sushiswap, Mooniswap utilizes a „virtual“ pricing mechanism which limits the value arbitrageurs can extract from liquidity providers. Over the course of a fixed period of time, buy and sell prices continually converge on their prices based on the „real“ underlying balance ratio. For example, a large buy order creates price movement which is not immediately reflected in the corresponding sell price. This new, better sell price is linearly converged on over the course of a fixed period of time. When a „sell“ order is placed within this time window, the price will be between the price before and after the original buy transaction. After a fixed period of time with no additional trading, the price is based exclusively on the real underlying balance ratio.



Comparison of Uniswap V2 LP income with potential Mooniswap LP income on different pools

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)

1. SafeMath.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/math/SafeMath.sol>

2. Math.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/math/Math.sol>

3. SafeERC20.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/SafeERC20.sol>

4. ERC20.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol>

5. IERC20.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/IERC20.sol>

6. EnumerableSet.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/EnumerableSet.sol>

7. Ownable.sol (0.6.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol>

8. ReentrancyGuard.sol (0.6.0)

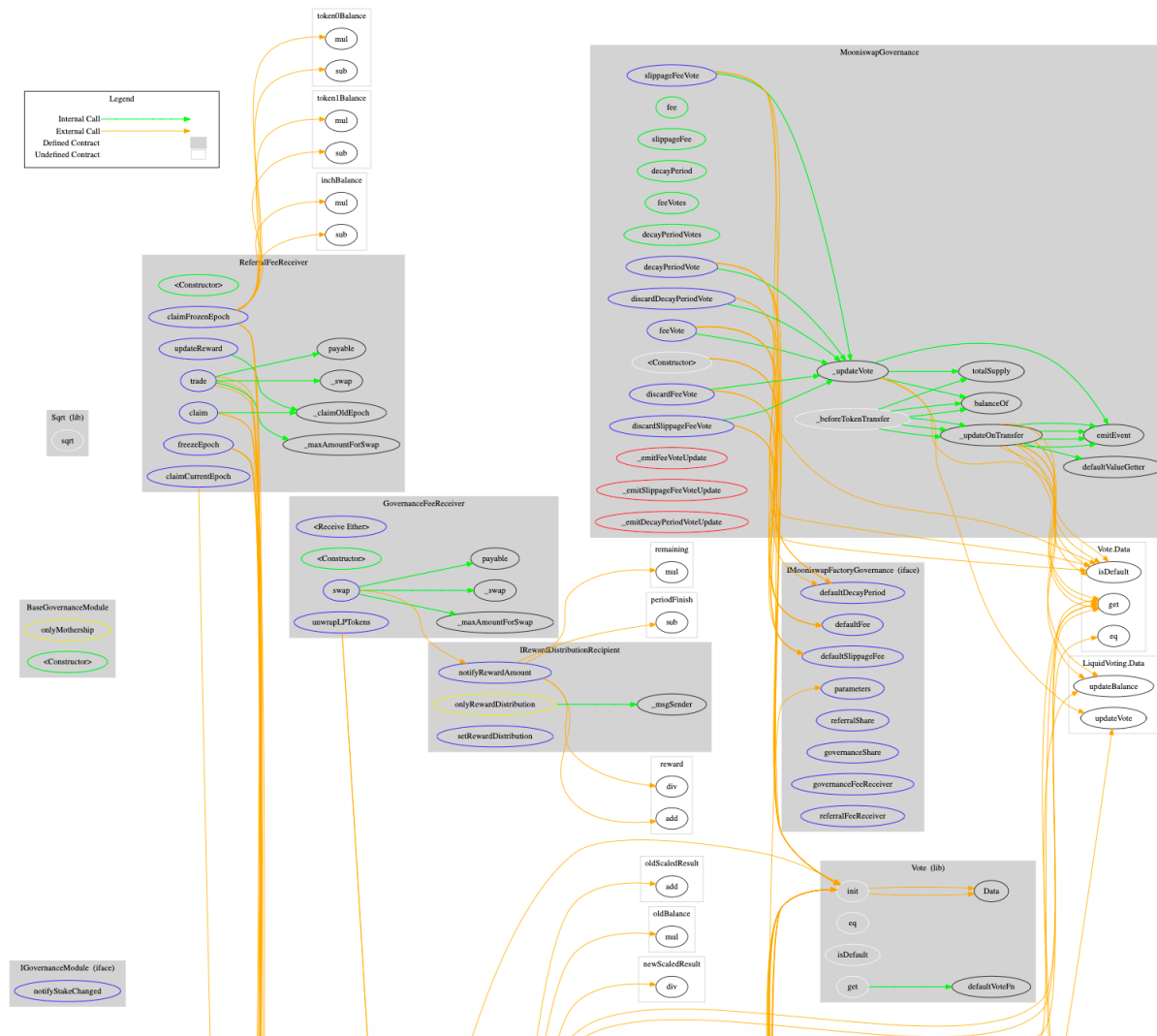
<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/ReentrancyGuard.sol>

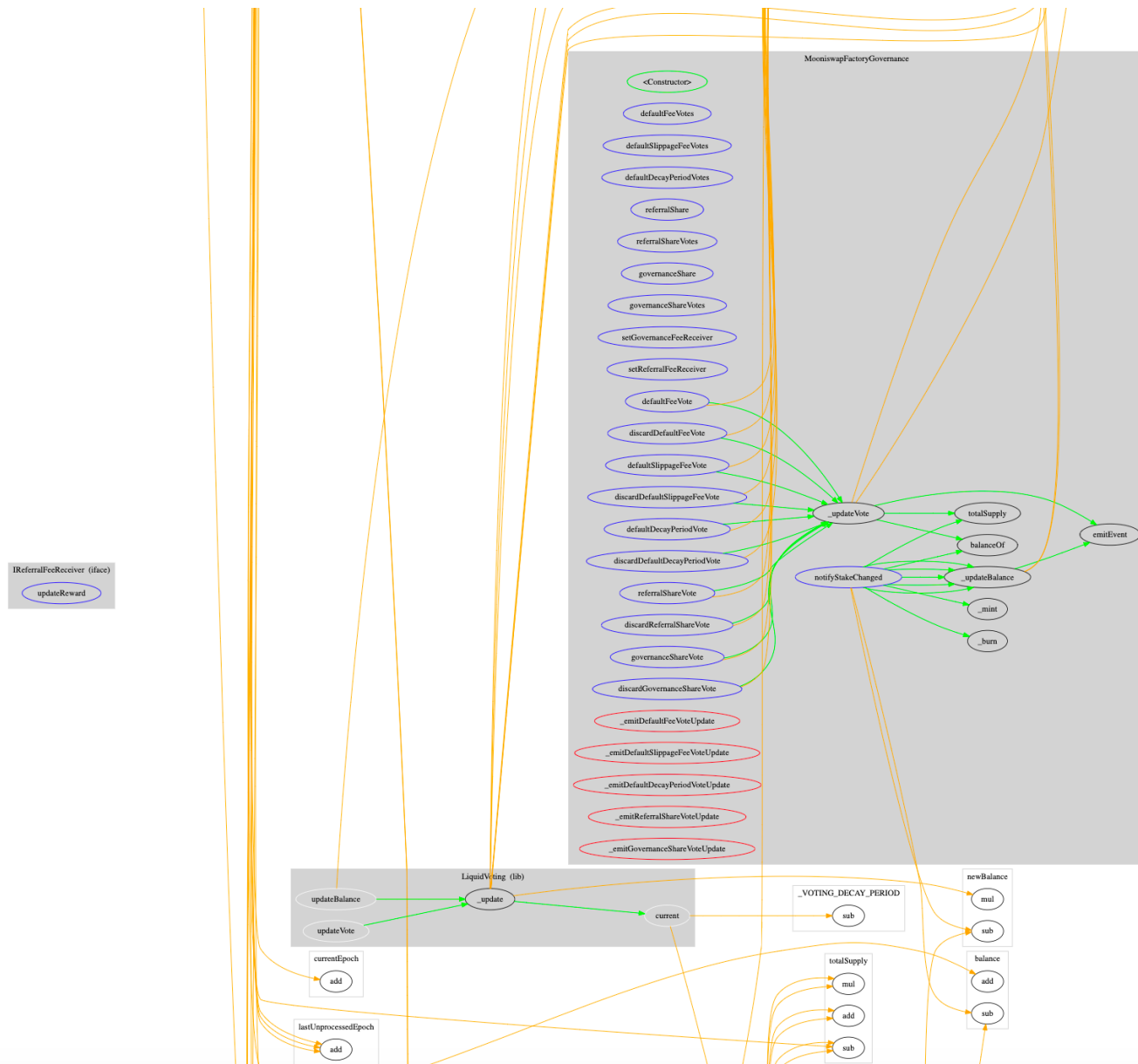
4.3 Tested Contract Files

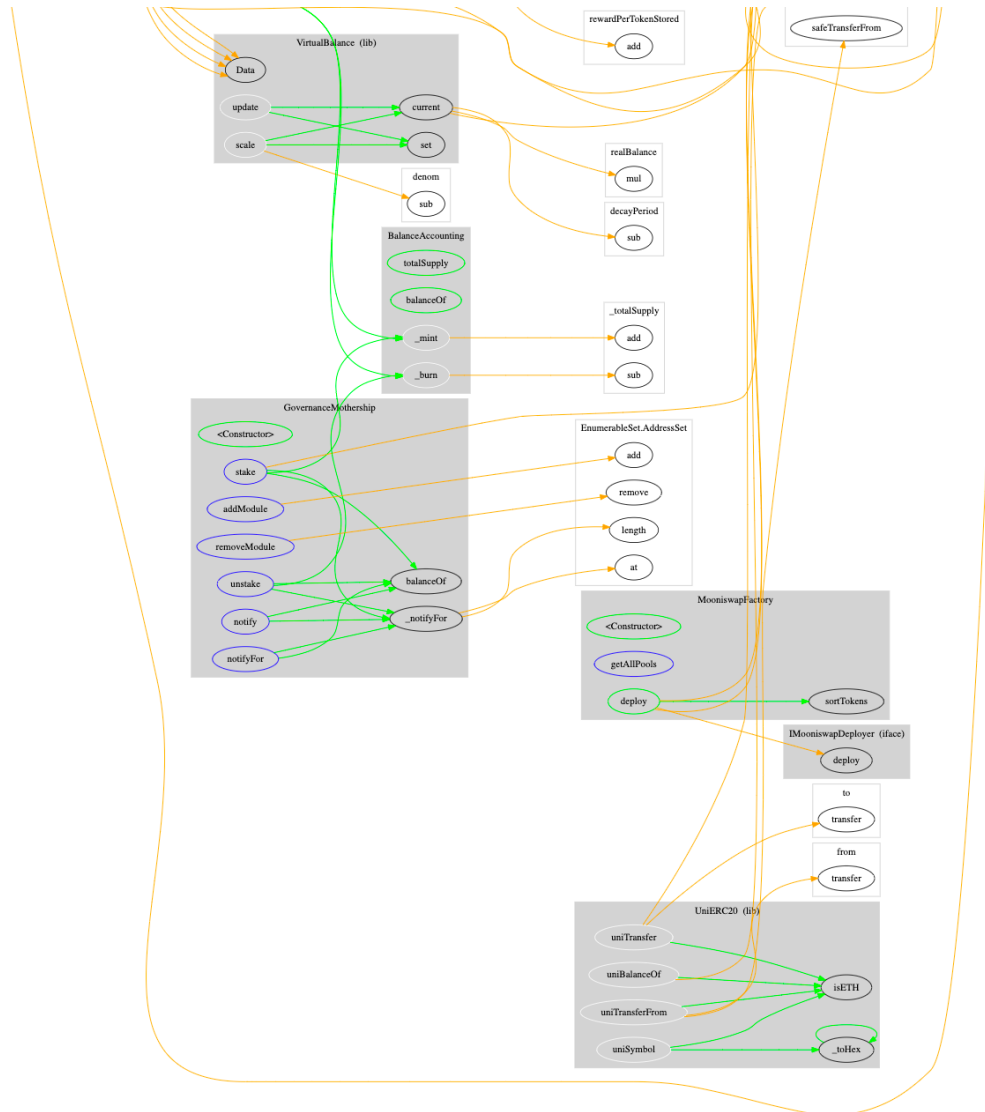
The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

File	Fingerprint (SHA256)
governance/BaseGovernanceModule.sol	3d7babad27c873657bd24570c774e9a4c34a8102a3f58ff6d5b88b58227d9287
governance/GovernanceFeeReceiver.sol	0309d841d21a5969d84ba966af089da0ab6cd631dc4beb96d82222f502a77920
governance/MooniswapFactoryGovernance.sol	7ad3f9f4e9cb6c25a8454b3574a8d020336b6897544aafc9f9be51fb3a768899e
governance/MooniswapGovernance.sol	903dd34d22341a73c9de80db4ad4692ad2da174d0d92fc45e63eadbbb8a9da5b
governance/rewards/IRewardDistributionRecipient.sol	5428e2f90c9319c9d80321fa26a6f7eb765604cd997ac2254ed0463248ebcc08
governance/rewards/Rewards.sol	e3e63608815efceae2cdcf01cdf576ef2128ca2b911983cb44df41132e21bf56
inch/GovernanceMothership.sol	b3ea21b1fff488890ed5b3b7591f67172cdf22c301489c8f01d6c2e6d3159b70
interfaces/IGovernanceModule.sol	e2d1e878f455164cdc37c4ef9dad77c7f2dd63a8e644bf85e5e31710459606ec
interfaces/IMooniswapDeployer.sol	5bfcfcf603b0c5b4a940979ae24c82bbe2206628ecc99c01e9b45ca853888aa9
interfaces/IMooniswapFactoryGovernance.sol	e08ff98c5f9f3d7b8ced65e3102c8f2d35e79b4b78ed8554dd8349aed6fe64cb
interfaces/IReferralFeeReceiver.sol	5e2b7b8c84137006f2d5b4fef8bf2c23db0c64ea1b01385f418f2feca6e8d1fc
libraries/LiquidVoting.sol	94d9e3d4db88df86cb0a225f1ab44bd2eeead45793f7fbf3f2e8d78703390c08
libraries/Sqrt.sol	a056706e9c27df78fa06fd213989b0ea53d48e34784170bc8e8fe31730cc3316
libraries/UniERC20.sol	f01c693bea4c5ad477c45714607b4e043aa7a53cf1d3241a9ca3e3574058577e
libraries/VirtualBalance.sol	1ea09c249f1f9705a4feec417d6391c07961cf12a1294e3cc5915cd15d3704e6
libraries/Vote.sol	5f2a706463f59bc8ec23aa9c12fd36c0adb98ad3cb6c38040e9fe4ace8ec5767
utils/BalanceAccounting.sol	24d2da011d58b0630ec03d3e511892718a314f2768df0166166a2e8efa779532
utils/Converter.sol	6dfc2b7c52613375f0f0d17e35fb80a008062f21aac0c058eac4bccbb95be23e
Mooniswap.sol	e261607eb99da4f2607058cf17447b352f73171050629b37b1d728ddabc810bc
MooniswapConstants.sol	d9a835219568bd9ad1c5a08d9b85589a2f14414fefd19f0a55a73c64bd5bef2b
MooniswapDeployer.sol	5ccb8836bd52275a4415508ca61bc82f518ff34b6565f1eff1ab5bf9f87138e1
MooniswapFactory.sol	56637b8c2294bf70aca7b92b92ff717c7273b74fd6041b66b7e533db51e2a5c5
ReferralFeeReceiver.sol	86d1554974072a3f7e455022aa49e947f2a4f05a59577a7769c6f666da49c209

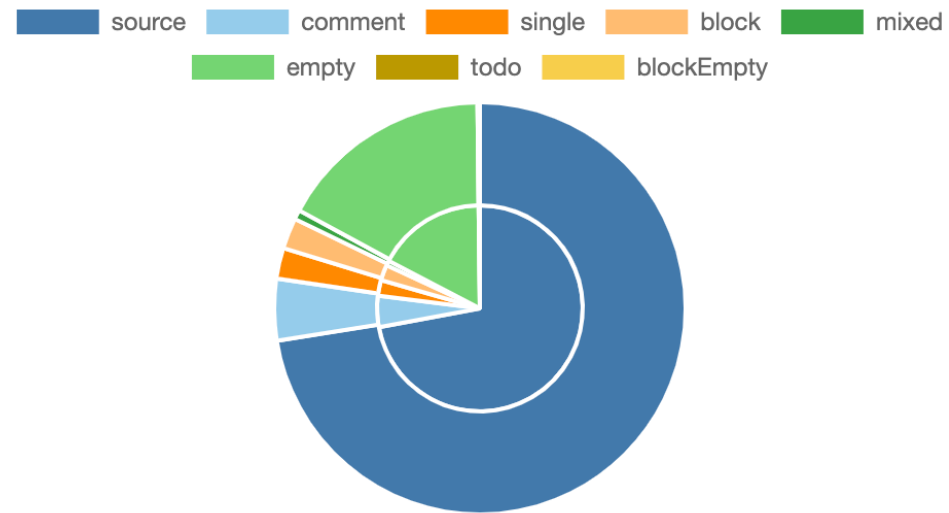
4.4 Metrics / CallGraph















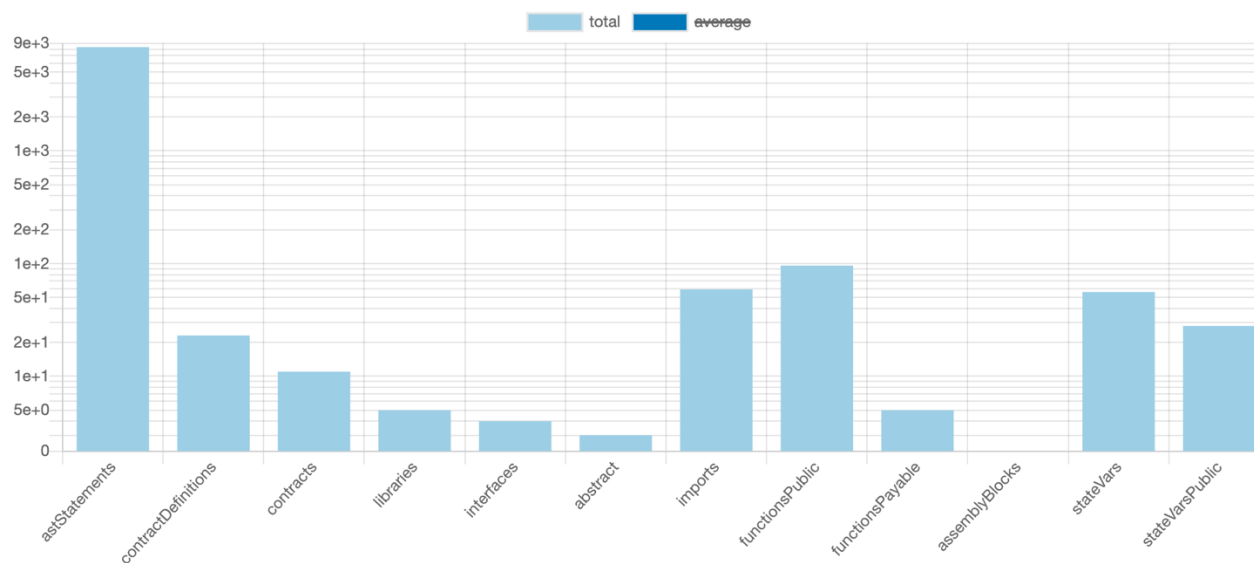


4.5 Metrics / Source Lines


































4.6 Metrics / Capabilities

Solidity Versions observed		 Experimental Features	 Can Receive Funds	 Uses Assembly	 Has Destroyable Contracts
<code>^0.6.0</code> <code>^0.6.12</code>			<code>yes</code>	<code>no</code> (0 asm blocks)	<code>no</code>
 Transfers ETH	 Low-Level Calls	 DelegateCall	 Uses Hash Functions	 ECTRecover	 New/Create/Create2
<code>yes</code>	<code>no</code>	<code>no</code>	<code>no</code>	<code>no</code>	<code>yes</code> → <code>NewContract: Mooniswap</code>



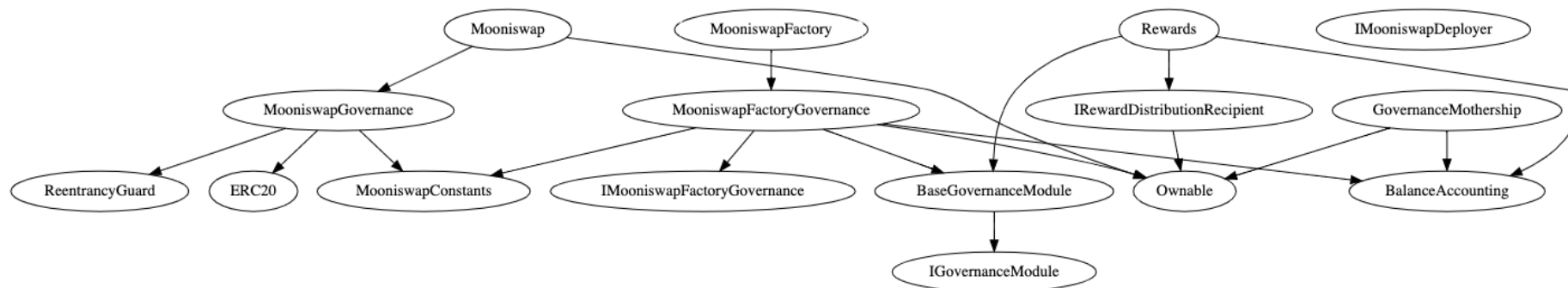
4.7 Metrics / Source Unites in Scope

Type	File	Logic Contracts	Interfaces	Lines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/Mooniswap.sol	1	_____	340	338	23	284	
	contracts/interfaces/IMooniswapDeployer.sol	_____	1	15	8	1	3	_____
	contracts/interfaces/IReferralFeeReceiver.sol	_____	1	8	7	1	3	_____
	contracts/interfaces/IGovernanceModule.sol	_____	1	8	7	1	3	_____
	contracts/interfaces/IMooniswapFactoryGovernance.sol	_____	1	16	7	1	17	_____
	contracts/governance/GovernanceFeeReceiver.sol	1	_____	30	30	2	39	
	contracts/governance/MooniswapGovernance.sol	1	_____	171	159	1	128	_____
	contracts/governance/MooniswapFactoryGovernance.sol	1	_____	228	214	1	144	_____
	contracts/MooniswapDeployer.sol	1	_____	25	19	1	16	
	contracts/governance/BaseGovernanceModule.sol	1	_____	20	20	1	7	_____
	contracts/MooniswapConstants.sol	1	_____	22	22	7	13	_____
	contracts/ReferralFeeReceiver.sol	1	_____	188	188	2	110	

Type	File	Logic Contracts	Interfaces	Lines	nSL OC	Comment Lines	Complex. Score	Capabilities
	contracts/MooniswapFactory.sol	1	_____	70	70	1	34	_____
	contracts/governance/rewards/IRewardDistributionRecipient.sol	1	_____	21	19	1	11	_____
	contracts/utills/Converter.sol	1	_____	108	108	2	90	
	contracts/utills/BalanceAccounting.sol	1	_____	31	31	1	11	_____
	contracts/libraries/Sqrt.sol	1	_____	23	23	2	4	_____
	contracts/libraries/LiquidVoting.sol	1	_____	92	67	1	31	_____
	contracts/libraries/VirtualBalance.sol	1	_____	38	38	1	19	_____
	contracts/libraries/UniERC20.sol	1	_____	107	107	2	80	
	contracts/libraries/Vote.sol	1	_____	44	44	1	6	_____
	contracts/governance/rewards/Rewards.sol	1	_____	107	102	1	76	_____
	contracts/inch/GovernanceMothership.sol	1	_____	67	67	1	52	_____
	Totals	19	4	1779	1695	56	1181	

MoonIsWapFactory.sol

Mooniswap.sol



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 Cannot pause / shutdown Mooniswap Factory

Severity: MEDIUM

Status: **FIXED** (<https://github.com/1inch-exchange/mooniswap-v2/commit/648157bfbe3ebaad1acf5de6df9a9add9a1ef448>)

File(s) affected: Mooniswap.sol

Author: [k06a](#)

Attack / Description	Code Snippet	Result/Recommendation
The Mooniswap.sol contract needs a mechanism for the owner to pause the contract. This disables the swap functionality. However, there is no corresponding mechanism to shutdown the contract. Consider introducing a mechanism for the owner to pause and unpause the contract. This will be important in case of emergency or fraudulent activities.	NA	We suggest to add a shutdown() method to the Mooniswap factory

5.1.2 Potential violation of Checks-Effects-Interaction pattern

Severity: MEDIUM

Status: **FIXED** (<https://github.com/1inch-exchange/mooniswap-v2/commit/961b4116f35a954c30435273da0cf6dba34cf053>)

File(s) affected: ReferralFeeReceiver.sol

Author: [ZumZoom](#)

Attack / Description	Code Snippet	Result/Recommendation
Potential violation of Checks-Effects-Interaction pattern. Could potentially lead to re-entrancy vulnerability.	<pre>Line: 10 contract ReferralFeeReceiver is IReferralFeeReceiver, Converter Line: 52 function freezeEpoch(Mooniswap mooniswap) external validPool(mooniswap) validSpread(mooniswap) { Line: 66 function trade(Mooniswap mooniswap, IERC20[] memory path) external validPool(mooniswap) validPath(path) Line: 129 function claimCurrentEpoch(Mooniswap mooniswap) external validPool(mooniswap) Line 141: function claimFrozenEpoch(Mooniswap mooniswap) external validPool(mooniswap)</pre>	<p>OpenZeppelin has it's own mutex implementation you can use called ReentrancyGuard. This library provides a modifier you can apply to any function called nonReentrant that guards the function with a mutex.</p> <p>View the source code for the OpenZeppelin ReentrancyGuard library here: https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/utils/ReentrancyGuard.sol</p> <p>Keep in mind that a nonReentrant function should be external. If another function calls the nonReentrant function it is no longer protected.</p>

LOW ISSUES

5.1.3 Potential front running attack or losing of allowance

Severity: LOW

Status: Acknowledged

File(s) affected: ERC20Permit.sol (1INCH Token)

Attack / Description	Code Snippet	Result/Recommendation
The _approve method replaces the allowance, so there are two potential problems here: 1. If a signer wants to increase the allowance from A to B , a receiver may withdraw A+B using the front-running attack. 2. If a signer wants to send A and B , but a receiver forgot to withdraw A , the receiver will lose ability to withdraw A .	Line: 53 _approve(owner, spender, amount);	We suggest to add permitIncrease, permitDecrease methods and use it instead of the permit .

5.1.4 Mismatch of argument name in _PERMIT_TYPEHASH and permit function

Severity: LOW

Status: **FIXED** (<https://github.com/1inch-exchange/1inch-token/commit/5332caa9b91403a022e74b49c9d0fc9c6d5419f4>)

File(s) affected: ERC20Permit.sol (1INCH Token)

Author: [ZumZoom](#)

Attack / Description	Code Snippet	Result/Recommendation
At the line ERC20Permit.sol#L27 the value argument is used but at the line ERC20Permit.sol#L32 the argument name is amount .	Line: 27 <pre>bytes32 private immutable _PERMIT_TYPEHASH = keccak256("Permit(address owner,address spender,uint256 value,uint256 nonce,uint256 deadline)");</pre> Line: 32 <pre>function permit(address owner, address spender, uint256 amount, uint256 deadline, uint8 v, bytes32 r, bytes32 s) public virtual override {</pre>	We suggest to rename amount to value in the permit function.

5.1.5 Use SafeCast library for uintXX/intXX wrapper

Severity: LOW

Status: **FIXED** (<https://github.com/1inch-exchange/mooniswap-v2/commit/6276231d02ded3c44863b23fc04a77356671108a>)

File(s) affected: MooniswapFactoryGovernance.sol, MooniswapGovernance.sol, ExplicitLiquidVoting.sol, LiquidVoting.sol, VirtualBalance.sol

Author: [ZumZoom](#)

Attack / Description	Code Snippet	Result/Recommendation
Wrappers over Solidity's uintXX/intXX casting operators with added overflow checks. Downcasting from uint256/int256 in Solidity does not revert on overflow. This can easily result in undesired exploitation or bugs, since developers usually assume that overflows raise errors. 'SafeCast' restores this intuition by reverting the transaction when such an operation overflows. Using this library instead of the unchecked operations eliminates an entire class of bugs, so it's recommended to use it always. Can be combined with {SafeMath} and {SignedSafeMath} to extend it to smaller types, by performing	Line: NA using SafeCast for uint256;	Consider to use SafeCast

all math on 'uint256' and 'int256' and then downcasting.		
--	--	--

INFORMATIONAL ISSUES

5.1.6 Missing natspec documentation

Severity: INFORMATIONAL

Status: Acknowledge

File(s) affected: all

Attack / Description	Code Snippet	Result/Recommendation
Solidity contracts can use a special form of comments to provide rich documentation for functions, return variables and more. This special form is named the Ethereum Natural Language Specification Format (NatSpec).	NA	<p>It is recommended to include natspec documentation and follow the doxygen style including @author, @title, @notice, @dev, @param, @return and make it easier to review and understand your smart contract.</p> <p>The team addressed the issue while auditing and added more documentation parts. https://github.com/1inch-exchange/mooniswap-v2/pull/13/commits</p>

5.1.7 Fix Spelling and Grammatical Errors

Severity: INFORMATIONAL

Status: **FIXED**

File(s) affected: ReferralFeeReceiver.sol

Attack / Description	Code Snippet	Result/Recommendation
Language mistakes were identified in the messages in the codebase. Fixing these mistakes can help improve the end-user experience by providing clear information on errors encountered, and improve the maintainability and auditability of the codebase.	Line 53: <code>require(info.lastUnprocessedEpoch == currentEpoch, "Previous epoch is not finlazed");</code> Line 138: <code>require(_userInfo.lastUnprocessedEpoch[mooniswap] == lastUnprocessedEpoch, "Epoch funcds already claimed");</code>	<code>require(info.lastUnprocessedEpoch == currentEpoch, "Previous epoch is not finalized");</code> <code>require(_userInfo.lastUnprocessedEpoch[mooniswap] == lastUnprocessedEpoch, "Epoch funds already claimed");</code>

5.1.8 A floating pragma is set

Severity: INFORMATIONAL

Code: SWC-103

Status: Acknowledge

File(s) affected: all

Attack / Description	Code Snippet	Result/Recommendation
The current pragma Solidity directive is <code>^0.6.12</code> ; It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.	Line: 1 <code>pragma solidity ^0.6.12;</code>	It is recommended to follow the example (0.6.12), as future compiler versions may handle certain language constructions in a way the developer did not foresee. Not effecting the overall contract functionality.

5.1.9 "dist:factory": "truffle-flattener Mooni*" (rename)

Severity: INFORMATIONAL

Code: NA

Status: **FIXED**

File(s) affected: packages.json

Attack / Description	Code Snippet	Result/Recommendation
Failed at the @1inch/mooniswap@0.0.1 dist:factory script.	Line: 43 "dist:factory": "truffle-flattener ./contracts/MooniFactory.sol awk '/SPDX-License-Identifier/ && c++>0 {next} 1' > ./MooniFactory.full.sol && solcjs --bin --abi --optimize ./MooniFactory.full.sol && mv ./__MooniFactory_full_sol_MooniFactory.bin ./MooniFactory.full.bin && mv ./__MooniFactory_full_sol_MooniFactory.abi ./MooniFactory.full.abi && rm ./*_sol_*" }}	Rename Mooni* to Mooniswap*

6. Executive Summary

The overall code quality of the project is very good, not overloaded with unnecessary functions and it is accompanied by unit tests, these is greatly benefiting the security of the contract. It correctly implemented widely-used and reviewed contracts from OpenZeppelin and for safe mathematical operations.

The main goal of the audit was to verify the claims regarding the security of the smart contract. During the audit, no critical or minor issues were found after the manual and automated security testing. The previous audits of mooniswap greatly benefiting the outcome, as the development team adopted the findings in the previous reports. It is recommended to include natspec documentation and follow the doxygen style including @author, @title, @notice, @dev, @param, @return and make it easier to review and understand your smart contract.

Previous Audits of Mooniswap in 2020

<https://mooniswap.exchange/docs/mooniswap-audit-report-2.pdf>

<https://mooniswap.exchange/docs/mooniswap-audit-report-3.pdf>

7. Deployed Smart Contract

VERIFIED

1INCH Token

[0x111111111117dc0aa78b770fa6a738034120c302](https://etherscan.io/address/0x111111111117dc0aa78b770fa6a738034120c302)

PENDING

MOONISWAP V2

0x00