ChainAudits

Smart Contract Audit

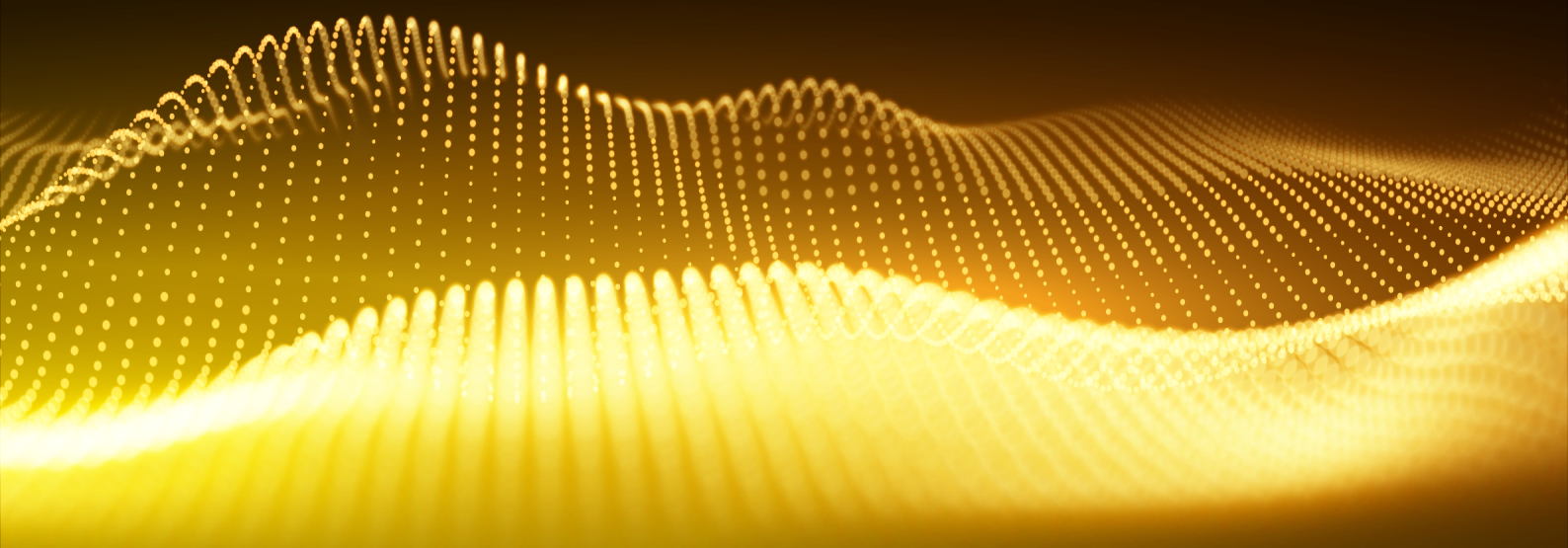
Security Assessment

Audit Dashboard Link

https://app.chainaudits.io/project/octoswap-1?cid=108

OctoSwap

V2-Periphery



ChainAudits

Table of Contents

ChainAudits

Project Details

CODEBASE
Upload

NETWORK
N/A

LANGUAGE
Solidity

NAME
OctoSwap

TOTAL SUPPLY
N/A

OWNER
N/A

WEBSITE
https://octo.exchange/

UNIT TESTS
N/A

FORK
N/A

ABOUT THE PROJECT
OctoSwap is a decentralized exchange on Monad that combines traditional automated market maker pools with concentrated liquidity pools in a single trading system. Its smart routing engine sources liquidity across both models to provide efficient execution for users. The platform includes swap functionality, liquidity management, and ecosystem features such as gasless approvals, MEV protections, incentives, and a points system.

Protocol Inheritance Analysis:

OctoSwap's contracts are verified to be directly forked from Uniswap contracts. All functional components were reviewed to confirm that the implemented logic matches Uniswap's original design. The differences identified are cosmetic in nature and do not alter the core functionality or expected behavior of the protocol.

Social Media Links

Linktree	N/A
Telegram	N/A
Facebook	N/A
Medium	N/A
X	• https://x.com/OctoSwapDex
Instagram	N/A
Youtube	N/A
Github	N/A
Reddit	N/A
Discord	N/A
Tiktok	N/A
LinkedIn	N/A
Flooz	N/A
Coingecko	N/A
Coinmarketcap	N/A

ChainAudits

Vulnerability Summary

Critical	Critical risks are those that affect the platform's safe operation and must be resolved before launch. Users should avoid investing in any project with unresolved critical risks.
High	High risks include centralization issues and logical errors. These risks can potentially result in the loss of funds or control over the project under certain conditions.
Medium	Medium risks might not directly threaten users' funds, but they can impact the platform's overall functionality.
Low	Low risks are similar to the above categories but on a smaller scale. They typically do not compromise the project's overall integrity but may lead to less efficient solutions.
Informational	Informational errors are recommendations aimed at improving code style or aligning operations with industry best practices. These usually do not affect the code's overall functionality.
Optimization	Optimization issues refer to inefficiencies in the code or smart contract logic that could be improved for better performance, reduced gas costs, or enhanced scalability. These issues do not impact security but can make the system more cost-effective and efficient.

Note - The provided audit report thoroughly examines the security aspects of the smart contract employed in the project, encompassing potential malicious manipulation of the contract's functions from external sources. However, it's important to note that this analysis does not incorporate functional or unit testing of the contract's logic. Therefore, we cannot ensure the absolute correctness of the contract's logic, including internal calculations within the formulae utilised in the contract.

ChainAudits

Overview

Score

The score is solely based on audited code findings and is not to be used as financial advice.

In-Scope Files

The team provided us with files to review during security audits. This audit covered the following files listed below with their respective SHA-1 Hash.

File Name	SHA-1 Hash
IERC20Metadata.sol	907ee3f5af9e65dd1c6d0fc44e4faee11f8204f6
IERC20PermitAllowed.sol	0f8ae33f339095b7745444ede48774f4023f7b0e
IERC721Permit.sol	ebae2281a1a29f8039994e45ae290dcc11cfd35e
IMulticall.sol	9e6b62357fe6d6748e7a5c4765c6ae6e1d732632
INonfungiblePositionManager.sol	131581a523b758568f390aace10b4aa66e2d8293
INonfungibleTokenPositionDescriptor.sol	1b2a07a417f71dd9b40e9fd376ad0ec00660c3ad
IPeripheryImmutableState.sol	f2c20d549085c951921ead205716daf43303aa78
IPeripheryPayments.sol	4ddd5c8d067b9e5d3aeb027bc39c980efc4dbe9d
IPeripheryPaymentsWithFee.sol	0da1ac6c52abfdbc8171d40f3ee898e08818cb31
IPoolInitializer.sol	5e91f53e858852ce1ce70f623f869c8976a0fe53
IQuoter.sol	414c29d48a547dd7d24c1b08382b8efee92ec420
IQuoterV2.sol	272b339467147ab0cf04fb477696353ca6a71f45
ISelfPermit.sol	fb8db7a56077ca32dd58a4a9bc25b54e2ad57071
ISwapRouter.sol	505cffd9f2e6df56e0009b776f9dd2f5523903b1
ITickLens.sol	5bb2a6b9e8f948f1d9ffb60f7d93ed7e72eecfd3
IV3Migrator.sol	c135ed05dd41de595b8a33bd98113bba4898cd49
IWETH9.sol	4d0d313953cb956315e444b9d94b8ccc27c3d99f
NonfungiblePositionManager.sol	6079c49bd1005107998e04103a02a837c175bbc4
NonfungibleTokenPositionDescriptor.sol	1a0a2ce96ae2263caff0251b09377f4443783118
OctoswapV2Router.sol	

3e455dbf016a5a3d60b62b9f175ac81ff173a64b

Please note that files with hash values different from those listed in this table have been changed after security checks, intentionally or unintentionally, as a particular hash value may indicate a changed state or potential vulnerabilities not checked in this scan.

> Note for Investors: We only examined agreements indicated in the indicated ratings. No contracts associated with the project beyond this range have been audited, therefore, we cannot provide insight or assume responsibility for their security

External/Public functions

External/public functions can be invoked outside of the contract, i.e., accessed by other contracts or external accounts on the blockchain. These functions are identified using external or public visibility modifiers in the function declaration.

State variables

State variables are stored on the blockchain as part of the contract conditions. They are declared at the contract level and can be accessed and changed by any action in the contract (except with modifiers like onlyOwner, etc.). State transitions can be described using a visibility modifier, such as public, private, or internal, which refers to the access to the transition.

Components

Contracts	3
Libraries	0
Interfaces	17
Abstract	0

Exposed Functions

Public	61
Payable	31
External	55
internal	22
Private	4
Pure	2
View	17

State Variables

Total	15
Public	2

Capabilities

Solidity Versions observed	=0.7.6, >=0.7.5, >=0.5.0, ^0.7.0
Transfers ETH	No
Can Receive Funds	yes
Uses Assembly	No
Has Destroyable Contracts	No

Audit Information

Strategies

While our projects undergo rigorous testing to address all known vulnerabilities, it's important to provide further clarification and understanding by outlining specific vulnerabilities that have been identified. This additional information enhances transparency and helps stakeholders comprehend the security measures taken and any potential risks mitigated.

Title	issue
Reentrancy	Improper Enforcement of Behavioral workflow
Unexpected Ether Balance	Improper Locking
Code with no effects	Irrelevant Code
Flash Loan Attacks	External Oracle Manipulation
Sandwich Attacks	A Form of Front Running
Write to arbitrary storage location	Improper Write to Arbitrary Storage Location
Variable Shadowing	Improper Coding Standards
Unprotected SELF-DESTRUCT	Improper Access Control
Potential Honeypot	Improper Function
Unprotected ETH withdrawal	Improper Access Control
Outdated Compiler Version	Using components with Known vulnerabilities
Weak Source of Randomness from Chain Attributes	Use of Insufficiently Random values
Unsafe use of libraries	Improper Implementation
Wrong Implementation of Token Standards	Improper Coding Standards

Auditing Strategy and Techniques Applied

All Projects at ChainAudits undergo our in-house developed “4-Eye” method. This process ensures that the code is analyzed not by a single auditor but by our entire technical team. Moreover, this is accomplished most efficiently, leaving no stone unturned.

We manually check every file, line by line. Automated tools are used solely to help us achieve faster and better results.

Code Analysis Methodology

The auditing process follows a routine series of steps:

- Manual Code Review:
  - Evaluate the overall structure and organization of the smart contract code line-by-line
  - Review the implementation of access control mechanisms to ensure that sensitive functions are appropriately restricted to authorized users.
  - Ensure that critical vulnerabilities are adequately tested and that edge cases and boundary conditions are covered.
- Static Code Analysis:
  - Utilise static analysis tools to scan the codebase for vulnerabilities and security issue
  - Identify common vulnerabilities like reentrancy, integer overflow/underflow, and unchecked external calls.
  - Evaluate compliance with coding standards and best practices, ensuring adherence t security guidelines.
- Code Structure and Architecture Review:
  - Analyze the overall structure and architecture of the smart contract code.
  - Assess the modularity, readability, and maintainability of the code.
  - Review the separation of concerns and adherence to design patterns for robustness and security.
- Security Best Practices Evaluation:
  - Evaluate the implementation of security best practices, including access control mech anisms and input validation.
  - Verify proper error handling to mitigate potential vulnerabilities and ensure contract robustness.
  - Check for gas optimization techniques to enhance efficiency and reduce transaction costs.
- External Dependency Assessment:
  - Assess the integration with external contracts, libraries, or services.
  - Review the security of external dependencies and their impact on the overall security of the smart contract codebase.
  - Ensure secure interactions with external components to prevent vulnerabilities and attack vectors.
- Post-Analysis Support:
  - Offer support and guidance to assist the development team in addressing identified vulnerabilities.
  - Collaborate with stakeholders to ensure effective implementation of recommended security enhancements.
  - Provide ongoing assistance and consultation to maintain the security of the smart contract codebase.

ChainAudits

General Findings - Type: ERC-20

Can the owner mint tokens arbitrarily?

The owner may have the ability to mint new tokens at any time, potentially diluting the token supply.

No

Can the owner burn tokens from any address without approval?

If the owner can burn tokens from user addresses without the holder's approval, it could lead to unjust token loss.

No

Can the owner pause or freeze token transfers?

The owner may have the authority to pause all transfers, which could prevent users from accessing their funds when needed.

No

Can the owner blacklist or whitelist addresses?

The owner may be able to control which addresses can hold or transfer tokens, potentially leading to censorship or arbitrary exclusion of participants.

No

Are there any integer overflow or underflow vulnerabilities?

The contract should ensure that all arithmetic operations are safe from overflow or underflow issues, especially if using older Solidity versions where these checks are not automatic.

No

Is there improper handling of Approvals?

The approve function should properly mitigate risks of double-spending when allowance is modified and then used in separate transactions.

No

Disclaimer

ChainAudits reports should not be construed as an endorsement or disapproval of any particular business or group. These reports do not reflect the economic value of any of the products or assets developed by the group. Also, ChainAudits does not consider integration with external contracts or services (e.g., Unicrypt, Uniswap, PancakeSwap).

ChainAudits reports aim to identify successful audit processes to help our clients improve the quality of the code and manage the risks associated with cryptographic tokens and blockchain technology. It is important to understand that blockchain technology and cryptographic assets pose significant ongoing risks. Each company and individual should conduct its own due diligence to maintain consistent safety measures. ChainAudits makes no representations about the security or performance of the technologies we audit.

ChainAudits does not provide any warranty or guarantee that the analyzed technology is completely defect-free, nor does it imply approval by the technology's owners. These audits should not be used to make input or output decisions; they will be involved in any project. They are not giving financial advice and should not be construed as such.