QuillAudits

Audit Report, September, 2024

For

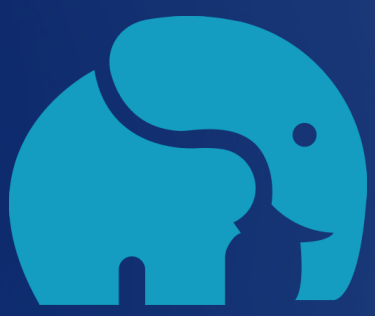


Table of Content

Executive Summary

Project Name	Stake DxPool
Project URL	https://stake.dxpool.com
Overview	Staked DX contracts allow for users to become an Ethereum validator through staking on its platform. There is a fee charge when this is set by the contract owner. With the aid of Openzeppelin Ownable and Pausable contract, some privileged functions in the BatchDeposit contract can only be caused by the contract owner and likewise be paused and unpaused. DxpoolBatchDeposit integrates the SSVNetwork and SSVToken, invoking most of the principal functions from the SSVNetwork while also allowing for batch deposit into the Beacon chain deposit contract.
Audit Scope	BatchDeposit DxpoolBatchDeposit
Contracts In-Scope	https://etherscan.io/address/0x4e089Dd33A1F71413c29e79839e79Cacfa341350 https://etherscan.io/address/0xe0bfacfc284db9496d856e287424df0bf2f56835
Language	Solidity
Blockchain	Ethereum
Method	Manual Analysis, Functional Testing, Automated Testing
Review 1	13th August 2024 - 10th September 2024
Updated Code Received	12th September 2024
Review 2	12th September 2024
Fixed In	https://etherscan.io/address/0xc1a1d2c713fffB68849390d492eCFeDaaD685334

Number of Issues per Severity

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	1	0	2

Checked Vulnerabilities

- Access Management
- Arbitrary write to storage
- Centralization of control
- Ether theft
- Improper or missing events
- Logical issues and flaws
- Arithmetic Correctness
- Race conditions/front running
- SWC Registry
- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Malicious libraries
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- ERC's conformance
- Dangerous strict equalities
- Tautology or contradiction
- Return values of low-level calls
- Missing Zero Address Validation
- Private modifier
- Revert/require functions
- Multiple Sends
- Using suicide
- Using delegatecall
- Upgradeable safety
- Using throw

Checked Vulnerabilities

- Using inline assembly
- Style guide violation
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
  Efficient use of gas.
  Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistic analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

Medium Severity Issues

1. withdrawSSVToken function allows for double spending of the SSVToken

Path

DxpoolBatchDeposit

Function

withdrawSSVToken

Description

It is possible for the withdrawAddress parameter passed into the withdrawSSVToken function will not be the same as the admin address. When this is the case, this function will give an allowance of the amount to the admin address who is the only privileged caller. Then, the same amount of token is sent to the withdrawAddress. The admin can spend this allowance, making for a double spending. Even if the admin does not spend the allowance, every withdrawal will continuously change the allowance given to the admin address.

Remediation

Remove the approval. This is not needed since the transfer of token from contract to recipient needs no approval.

Status

Resolved

Informational Issues

2. State variable should be marked as immutable

Path

BatchDeposit

Function

officialDepositContract

Description

This variable was only set at the constructor and there are no functions to further update this variable.

Recommendation

Making the variable immutable will prevent it from taking up a storage slot.

Status

Resolved

3. Use call instead of transfer function to send ether

Path

BatchDeposit

Function

withdrawFee

Description

Fees accumulated over time can be withdrawn with this function but this function uses the transfer function for sending ether. The transfer function imposes 2300 fixed gas on the recipient code if it is a contract, like the send function.

Recommendation

Using the call function is flexible and performant if the recipient is a contract address.

Status

Resolved

Reference

https://solidity-by-example.org/sending-ether/

Functional Tests Cases

Some of the tests performed are mentioned below:

- Should allow the contract owner to set the fee and update max validators
- Should successful call the deposit function with all parameters
- Should reverts when inaccurate amount is sent with the deposit function
- Should allow contract to be paused and unpaused by the contract owner
- Should call principal functions from the SSVNetwork
- Should successful deposit ether into the Beacon chain deposit contract
- Should test for withdrawal of ether and SSVToken in the DxpoolBatchDeposit contract

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of StakedDX. We performed our audit according to the procedure described above.

Some issues of medium and informational severity were found. Some suggestions, gas optimizations and best practices are also provided in order to improve the code quality and security posture. In The End, StakedDX Team Resolved all Issues.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in StakedDX. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of StakedDX. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of StakedDX Team to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With six years of expertise, we've secured over 1000 projects globally, averting over $30 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

1000+
Audits Completed

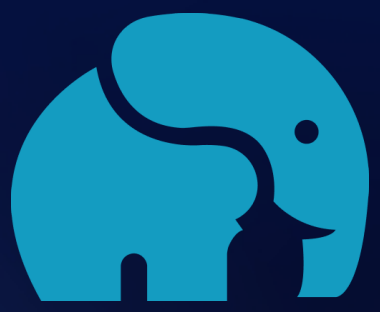
$30B
Secured

1M+
Lines of Code Audited

Follow Our Journey

Audit Report
September, 2024

For



QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com