SOLIDProof
Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC Development | Marketing

MADE IN GERMANY

Zkswap Finance

AUDIT

SECURITY ASSESSMENT

18. March, 2024

FOR

Introduction 3
Disclaimer 3
Project Overview 4
Summary 4
Social Medias 5
Audit Summary 6
File Overview 7
Imported packages 8
Components 9
Exposed Functions 9
Capabilities 10
Inheritance Graph 11
Audit Information 12
Vulnerability & Risk Level 12
Auditing Strategy and Techniques Applied 13
Methodology 13
Overall Security 14
Upgradeability 14
Ownership 15
Ownership Privileges 16
Minting tokens 16
Burning tokens 17
Blacklist addresses 18
Fees and Tax 19
Lock User Funds 20
Centralization Privileges 21
Audit Results 23

Introduction

SolidProof.io is a brand of the officially registered company MAKE Network GmbH, based in Germany. We're mainly focused on Blockchain Security such as Smart Contract Audits and KYC verification for project teams.
Solidproof.io assess potential security issues in the smart contracts implementations, review for potential inconsistencies between the code base and the whitepaper/documentation, and provide suggestions for improvement.

Disclaimer

SolidProof.io reports are not, nor should be considered, an "endorsement"or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc'...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof's position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of the security or functionality of the technology we agree to analyze.

Project Overview

Summary

Project Name	Zkswap Finance
Website	https://zkswap.finance/
About the project	zkSwap Finance is the top DEX and the first Swap to Earn DeFi Platform on zkSync Era ecosystem, pioneering a unique incentive model that rewards both liquidity providers and traders.
Chain	ZK Sync
Language	Solidity
Codebase	https://github.com/ZkSwapFinance/stableswap-contracts
Forked Status	This project is 1:1 forked with Pancake stable swap contracts. Here, are the links of the contracts: https://bscscan.com/address/0x25a55f9f2279a54951133d503490342b50e5cd15#code https://bscscan.com/address/0x1179ADfa22dD0e5050C1c00C9f8543A77F75A2c0#code https://bscscan.com/address/0xee1bcc9F1692E81A281b3a302a4b67890BA4be76#code https://bscscan.com/address/0xdDFCDAacC836dd5A1AE2D375fFb153cE59DD09ff#code https://bscscan.com/address/0xc54d35a8Cfd9f6dAe50945Df27A91C9911A03ab1#code https://bscscan.com/address/0x150c8AbEB487137acCC541925408e73b92F39A50#code https://bscscan.com/address/0x6AF7a605953C0B462EE9540217F5bD80878c6b2E#code https://bscscan.com/address/0x13f4EA83D0bd40E75C8222255bc855a974568Dd4#code
Deployed contracts	StableSwapLPFactory: 0xD57B17609055168be50e48D27DBB8d97Be2FF90C StableSwapLP: 0xe0e4Fa0a608e8C22d5a7e411e53a9F2B54784241 StableSwapTwoPoolDeployer: 0xFe26d4B82F03252901B55eD05FC9d42B53fe03d7 StableSwapThreePoolDeployer: 0x9798b5E3a38dd51e32695F4406Bd008B99e22e1a StableSwapFactory: 0xe35f6725CD640d3913a7999285d48C01Cb20697F

	StableSwapTwoPoolInfo: 0x1Be7b2A61ABd739F60D0D36104d480ea8763705D StableSwapThreePoolInfo: 0x82d4fC894d8F1b69B9867CaEd29e7B1Fa5b3689c StableSwapThreePool: 0xa057BddEcD61f9E2C99C74D976cAe7c8bD60717E StableSwapRouter: 0x04FD05DDA266791f7aDdEC2A4DF06b2A676eD12A
Commit	https://github.com/ZkSwapFinance/stableswap-contracts/commit/e8dfd0519df666e6676c83b8820ba7f4f333a862
Unit Tests	Not Provided

Social Medias

Telegram	https://t.me/zkSwap_Finance_Official
Twitter	https://twitter.com/zkSwap_finance
Facebook	N/A
Instagram	N/A
GitHub	https://github.com/ZkSwapFinance
Reddit	N/A
Medium	https://medium.com/@zkswap_finance
Discord	https://discord.com/invite/4eHMumaJDA
YouTube	https://www.youtube.com/@zkSwap_finance
TikTok	https://www.tiktok.com/@zkswap_finance
LinkedIn	N/A

Audit Summary

Version	Delivery Date	Change Log
v1.0	06. March 2024	· Layout Project · Automated/ Manual-Security Testing · Summary
v1.1	18. March 2024	· Reaudit

Note – The following audit report presents a comprehensive security analysis of the smart contract utilized in the project that includes outside manipulation of the contract's functions in a malicious way. This analysis did not include functional testing (or unit testing) of the contract/s logic. We cannot guarantee 100% logical correctness of the contract as we did not functionally test it. This includes internal calculations in the formulae used in the contract.

File Overview

The Team provided us with the files that should be tested in the security assessment. This audit covered the following files listed below with an SHA-1 Hash.

File Name	SHA-1 Hash
contracts\StableSwapTwoPoolDeployer.sol	f2ca4dca03fc03049f60daa027e775b3c7c9240d
contracts\StableSwapTwoPoolInfo.sol	51af31d87b2c17b202ece8289a63c5a2d8cbb09a
contracts\StableSwapTwoPool.sol	0189e7089c57107ebe1a9e6f4e857d5c17e5a0b5
contracts\StableSwapThreePoolInfo.sol	1bbf3bb8b52448f7a885f8d10a3af27ce740c5fd
contracts\StableSwapThreePoolDeployer.sol	633daadc28f2d1a58f3d9dbdffe0d13bddaea3a0
contracts\StableSwapThreePool.sol	74d2d98fac08e666fac7b0799657534faaad0535
contracts\StableSwapRouter.sol	f88ce610ed2d739bc6102754b4e512ba6e7f9d30
contracts\StableSwapLPFactory.sol	cc94fcda3cfc83c4a0b468001586b2b8398f0c10
contracts\StableSwapLP.sol	dadc7104d5b48d6d4efdf1ffc2669ad147fb2f6b
contracts\StableSwapFactory.sol	7caa662e7fa16f4b8d86443c7a2c6a9b4c555854
contracts\libraries\TransferHelper.sol	3d8137cdc66c9a6cd404b9420cad3d600d61eacf
contracts\interfaces\IStableSwapRouter.sol	a9545a7dd1dcd809ff9704f405b2eb6cb3a3b022
contracts\interfaces\IStableSwapLPFactory.sol	faf5339415af07610ad90ad3f44f38d963376bea
contracts\interfaces\IStableSwapLP.sol	b0c2b902ec21f7112bba6481bd38550e364fbd2e
contracts\interfaces\IStableSwapInfo.sol	663ad5133c414df2cbd99bb45c820d35cadb4bca
contracts\interfaces\IStableSwapFactory.sol	3eb2077d46dedc69c062aac07c977029d3fa6e9c
contracts\interfaces\IStableSwapDeployer.sol	1c31d895792712c08f9a34e70112f188411021fb
contracts\interfaces\IStableSwap.sol	7378e996fff8d4058a1d95eb1233cb4e5ffaed67

Please note: Files with a different hash value than in this table have been modified after the security check, either intentionally or unintentionally. A different hash value may (but need not) be an indication of a changed state or potential vulnerability that was not the subject of this scan.

Imported packages.

Used code from other Frameworks/Smart Contracts.

Dependency / Import Path	Count
@openzeppelin/contracts/access/Ownable.sol	6
@openzeppelin/contracts/security/ReentrancyGuard.sol	3
@openzeppelin/contracts/token/ERC20/ERC20.sol	1
@openzeppelin/contracts/token/ERC20/IERC20.sol	1
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	2
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	4

Note for Investors: We only audited contracts mentioned in the scope above. All contracts related to the project apart from that are not a part of the audit, and we cannot comment on its security and are not responsible for it in any way.

External/Public functions

External/public functions are functions that can be called from outside of a contract, i.e., they can be accessed by other contracts or external accounts on the blockchain. These functions are specified using the function declaration's external or public visibility modifier.

State variables

State variables are variables that are stored on the blockchain as part of the contract'sstate. They are declared at the contract level and can be accessed and modified by any function within the contract. State variables can be needed within visibility modifier, such as public, private or internal, which determines the access level of the variable.

Components

📝 Contracts	📚 Libraries	🔍 Interfaces	🎨 Abstract
10	1	7	0

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

🌐 Public	💰 Payable
113	1

External	Internal	Private	Pure	View
100	102	1	12	83

StateVariables

Total	🌐 Public
74	72

Capabilities

Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
0.8.16	----------	Yes	yes (3 asm blocks)	----------

Transfers ETH	Low-Level Calls	Delegate Call	Uses Hash Functions	ECRecover	New/Create/Create2
			Yes		yes → AssemblyCall:Name:create2

Inheritance Graph

An inheritance graph is a graphical representation of the inheritance hierarchy among contracts. In object-oriented programming, inheritance is a mechanism that allows one class (or contract, in the case of Solidity) to inherit properties and methodsfrom another class. It shows the relationships between different contracts and how they are related to each other through inheritance.

Audit Information

Vulnerability & Risk Level

Risk represents the probability that a certain source threat will exploit the vulnerability and the impact of that event on the organization or system.The 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 aspossible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executingthe 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

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to check the repository for security-related issues, code quality, and compliance with specifications and best practices. To this end, our team of experienced pen-testers andsmart contract developers reviewed the code line by line and documented any issues discovered.

We check every file manually. We use automated tools only so that they help us achieve faster and better results.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
   a. Reviewing the specifications, sources, and instructions provided to
   SolidProof to ensure we understand the size, scope, and functionality of the
   smart contract.
   b. Manual review of the code, i.e., reading the source code line by line to identify potential vulnerabilities.
   c. Comparison to the specification, i.e., verifying that the code does what is described in the specifications, sources, and instructions provided to SolidProof.

2. Testing and automated analysis that includes the following:
   a. Test coverage analysis determines whether test cases cover code and how much code is executed when those test cases are executed.
   b. Symbolic execution, which is analysing a program to determine what inputs cause each part of a program to execute.

3. Review best practices, i.e., review smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on best practices, recommendations, and research from industry and academia.

4. Concrete, itemized and actionable recommendations to help you secure your smart contracts.

Overall Security

Upgradeability

Contract is not an upgradable	✅ **Deployer cannot update the contract with new functionalities.**
Description	The contract is not an upgradeable contract. The Deployer is not able to change or add any functionalities to the contract after deploying.
Comment	N/A

Ownership

Contract ownership is not renounced.	❌ **The ownership is not renounced.**
Description	The owner has not renounced the ownership that means that the owner retains control over the contract's operations, including the ability to execute functions that may impact the contract's users or stakeholders. This can lead to several potential issues, including: • Centralizations • The owner has significant control over contract's operations.
Comment	N/A

Note – *The contract cannot be considered as renounced till it is not deployed or having some functionality that can change the state of the contract.*

Ownership Privileges

These functions can be dangerous. Please note that abuse can lead to financial loss. We have a guide where you can learn more about these Functions.

Minting tokens

Minting tokens refer to the process of creating new tokens in a cryptocurrency or blockchain network. This process is typically performed by the project's owner or designated authority, who has the ability to add new tokens to the network's total supply.

Contract owner cannot mint new tokens.	✅ **The owner cannot mint new tokens.**
Description	The owner is able to mint new tokens once the contract is deployed.
Comment	The logic of the AMM contracts only permits the StableSwapTwoPool/StableSwapThreePool contract to act as the sole minter.

Burning tokens

Burning tokens is the process of permanently destroying a certain number of tokens, reducing the total supply of a cryptocurrency or token. This is usually done to increase the value of the remaining tokens, as the reduced supply can create scarcity and potentially drive up demand.

Contract owner cannot burn tokens	✅ **The owner cannot burn tokens.**
Description	The owner is able burn tokens without any allowances.
Comment	The logic of the AMM contracts only permits the StableSwapTwoPool/StableSwapThreePool contract to act as the sole minter. This is standard for AMMs, where the liquidity pool manages the minting and burning of the liquidity (LP) token.

Blacklist addresses

Blacklisting addresses in smart contracts is the process of adding a certain address to a blacklist, effectively preventing them from accessing or participating in certain functionalities or transactions within the contract. This can be useful in preventing fraudulent or malicious activities, such as hacking attempts or money laundering.

Contract owner cannot blacklist addresses.	✅ **The owner cannot blacklist wallets.**
Description	The owner cannot blacklist wallets from transferring of tokens.
Comment	N/A

Fees and Tax

In some smart contracts, the owner or creator of the contract can set fees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort indeveloping and maintaining the contract.

Contract owner cannot set fees more than	✅ **The owner cannot set fees more than 25%.**
Description	The owner can set fees of more than 25%.
Comment	The fees in the contract cannot exceeds to more than 10%.

Lock User Funds

In a smart contract, locking refers to the process of restricting access to certain tokens or assets for a specified period of time. When token or assets are locked in a smart contract, they cannot be transferred or used until the lock-up period has expired or certain conditions have been met.

Contract owner cannot lock funds.	✅ **The owner can lock user funds.**
Description	The owner can lock the functionalities in the contract for an indefinite period of time.
Comment	The contract contains the pausable functionality which can lock the functions The pause function (L695-698) only affects 4 functions where hackers might exploit: exchange (L410), add_liquidity (L253), remove_liquidity_imbalance (L461), remove_liquidity_one_coin (L604). This pause function is implemented to enhance the security of the contract by reducing potential continuous losses to users' assets in the event of a hack. During the pause, users can still freely withdraw their assets using the remove_liquidity function.

Centralization Privileges

Centralization can arise when one or more parties have privileged access or control over the contract's functionality, data, or decision-making. This can occur, for example,if the contract is controlled by a single entity or if certain participants have special permissions or abilities that others do not.

In the project, there are authorities that have access to the following functions:

File	Privileges
StableSwapFactory	➢ The owner can create a swap pair in the contract. ➢ The owner can create a pool pair in the contract. ➢ The owner can add info to the contract.
StableSwapThreePoolDeployer	➢ The owner can create the swap pair in the contract.
StableSwapThreePool	➢ The stable swap factory contract can initialize the contract. ➢ The owner can add a ramp in the contract. ➢ The owner can stop the ramp in the contract. ➢ The owner can update the fees not more than 1e9 in the contract. ➢ The owner can update the protocol fees of not more than 1e10 in the contract. ➢ The owner can apply new fees. ➢ The owner can revert to new parameters. ➢ The owner can withdraw the protocol fees from the contract. ➢ The owner can donate the protocol fees from the contract. ➢ The owner can pause/un-pause the functionalities in the contract.
StableSwapLPFactory	➢ The owner can create a swap liquidity pool in the contract.
StableSwapLP	➢ The owner can add minters to the contract. ➢ The minter can mint tokens. ➢ The owner can burn tokens from other wallets without allowance.

StableSwapTwoPoolDeployer
StableSwapTwoPool

- The owner can create a swap pair in the contract.
- The stable swap factory contract can initialize the contract.
- The owner can add a ramp in the contract.
- The owner can stop the ramp in the contract.
- The owner can update the fees not more than 1e9 in the contract.
- The owner can update the protocol fees of not more than 1e10 in the contract.
- The owner can apply new fees.
- The owner can revert to new parameters.
- The owner can withdraw the protocol fees from the contract.
- The owner can donate the protocol fees from the contract.
- The owner can pause/un-pause the functionalities in the contract.

Recommendations

To avoid potential hacking risks, it is advisable for the client to manage the private key of the privileged account with care. Additionally, we recommend enhancing the security practices of centralized privileges or roles in the protocol through a decentralized mechanism or smart-contract-based accounts, such as multi-signature wallets.

Here are some suggestions of what the client can do:

- Consider using multi-signature wallets: Multi-signature wallets require multiple parties to sign off on a transaction before it can be executed, providing an extra layer of security e.g. Gnosis Safe
- Use of a timelock at least with a latency of e.g. 48-72 hours for awareness of privileged operations
- Introduce a DAO/Governance/Voting module to increase transparency and user involvement
- Consider Renouncing the ownership so that the owner cannot modify any state variables of the contract anymore. Make sure to set up everything before renouncing.

Audit Result

Critical Issues

No critical issues

High Issues

No high issues

Medium Issue

No medium issues

Low Issue

#1 | Missing zero or dead address check.

File	Severity	Location	Status
StableSwapFactory	Low	L185-193	Fixed

Description – It is recommended to check that the address cannot be set to zero or dead address.

Informational Issue

#1 | NatSpec Documentation missing

File	Severity	Location	Status
All	Informational	--	ACK

Description – If you started to comment on your code, also comment on all other functions, variables, etc.

Legend for the Issue Status

Attribute or Symbol	Meaning
Open	The issue is not fixed by the project team.
Fixed	The issue is fixed by the project team.
Acknowledged(ACK)	The issue has been acknowledged or declared as part of business logic.

Blockchain Security | Smart Contract Audits | KYC Development | Marketing

MADE IN GERMANY